BigData&DataAnalytics

Machine learning techniques to the problem of indoor locating

Mauricio Carvajal
IOT ANALYTICS

Table of contents

1. Description of data science process

Before starting the project, we follow the data science work flow in order to have a better understanding of the data that we are going to work on and provide with better and meanfully insights.

So the first start is to define the problem statement, and the goal of this project, then where the data is taken, the description of the data and how we are going to handle the data.

Handling this a lot of information brings a lot of challenges, so before starting is needed to analysis the risk of those issues finds ways to address and overcome those problems.

1.1 Statements of the investigation.

The investigation is following the need for the company to review if is feasible the develop of a system that helps for people to navigate in a complex, unfamiliar interior space without getting lost meanwhile the GPS works.

1.1.1 Problem Statement

The IOT Analytics doesn't have any solution that can use wifi finger printing for indoors locations.

1.1.2 Goal

The IOT Analytics has the task to evaluate if develop a solution that enables users to navigate in complex indoors spaces using WIFI fingerprinting is feasible.

1.2 Description and location of related data sources

For develop the model, there data is taking from the following link:

http://archive.ics.uci.edu/ml/datasets/UJIIndoorLoc

1.2.1 Data Set Information

Many real-world applications need to know the localization of a user in the world to provide their services. Therefore, automatic user localization has been a hot research topic in the last years. Automatic user localization consists of estimating the position of the user (latitude, longitude and altitude) by using an electronic device, usually a mobile phone. Outdoor localization problem can be solved very accurately thanks to the inclusion of GPS sensors into the mobile devices. However, indoor localization is still an open problem mainly due to the loss of GPS signal in indoor environments. Although, there are some indoor positioning technologies and methodologies, this database is focused on WLAN fingerprint-based ones (also know as WiFi Fingerprinting).

The UJIIndoorLoc database covers three buildings of Universitat Jaume I with 4 or more floors and almost 110.000m2. It can be used for classification, e.g. actual building and floor identification, or regression, e.g. actual longitude and latitude estimation. It was created in 2013 by means of more than 20 different users and 25 Android devices. The database consists of 19937 training/reference records (trainingData.csv file) and 1111 validation/test records (validationData.csv file).

The 529 attributes contain the WiFi fingerprint, the coordinates where it was taken, and other useful information.

Each WiFi fingerprint can be characterized by the detected Wireless Access Points (WAPs) and the corresponding Received Signal Strength Intensity (RSSI). The intensity values are represented as negative integer values ranging -104dBm (extremely poor signal) to 0dbM. The positive value 100 is used to denote when a WAP was not detected. During the database creation, 520 different WAPs were detected. Thus, the WiFi fingerprint is composed by 520 intensity values.

Then the coordinates (latitude, longitude, floor) and Building ID are provided as the attributes to be predicted.

1.2.2 Additional information

The particular space (offices, labs, etc.) and the relative position (inside/outside the space) where the capture was taken have been recorded. Outside means that the capture was taken in front of the door of the space.

Information about who (user), how (android device & version) and when (timestamp) WiFi capture was taken is also recorded.

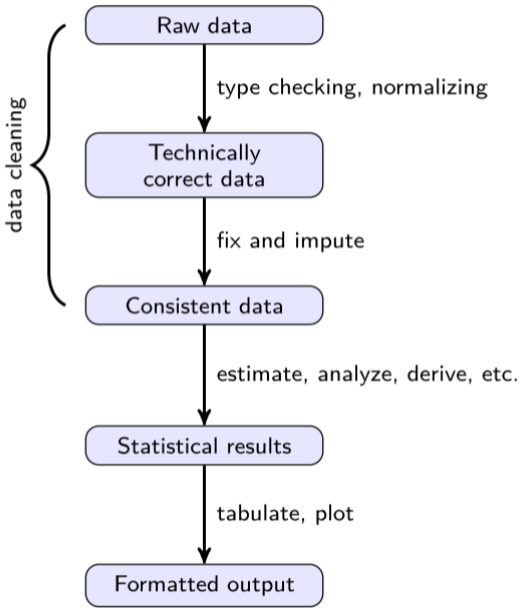
1.2.3 Attribute Information

Atribute ID	*Attribute Name*	*Attribute Description*	*Description of the data*
Attribute 001	WAP001	Intensity value for WAP001	Negative integer values from -104 to 0 and +100. Positive value 100 used if WAP001 was not detected.
Attribute 520	WAP520	Intensity value for WAP520	Negative integer values from -104 to 0 and +100. Positive Vvalue 100 used if WAP520 was not detected.
Attribute 521	Longitude	Longitude	Negative real values from -7695.9387549299299000 to -7299.786516730871000
Attribute 522	Latitude	Latitude	Positive real values from 4864745.7450159714 to 4865017.3646842018.
Attribute 523	Floor	Altitude in floors inside the building	Integer values from 0 to 4.
Attribute 524	BuildingID	ID to identify the building	Categorical integer values from 0 to 2.
Attribute 525	SpaceID	ID number to identify the Space (office, corridor, classroom) where the capture was taken	Categorical integer values.
Attribute 526	RelativePosition	Relative position with respect to the Space (1 - Inside, 2 - Outside in Front of the door)	Categorical integer values.
Attribute 527	UserID	User identifier	Categorical integer values.
Attribute 528	PhoneID	Android device identifier	Categorical integer values.
Attribute 529	Timestamp	UNIX Time when the capture was taken	Integer value.

1.2 How you will manage the data for the project

1.3.1 Preprocessing data

For the following project, we will use the framework to manage the data, in which the first step is obtain the data, as raw data and then to preprocessing.



The first state (Raw data) is the data as it comes in. Raw data files may lack headers, contain wrong data types (e.g. numbers stored as strings), wrong category labels, unknown or unexpected character encoding and so on. In short, reading such files into an R data.frame directly is either difficult or impossible without some sort of preprocessing. Once this preprocessing has taken place, data can be deemed Technically correct. That is, in this state data can be read into an R data.frame, with correct names, types and labels, without further trouble. However, that does not mean that the values are error-free or complete. For example, an age variable may be reported negative, an under-aged person may be registered to possess a driver's license, or data may simply be missing. Such inconsistencies obviously depend on the subject matter that the data pertains to, and they should be ironed out before valid statistical inference from such data can be produced. Consistent data is the stage where data is ready for statistical inference. It is the data that most statistical theories use as a starting point. Ideally, such theories can still be applied without taking previous data cleaning steps into account. In practice however, data cleaning methods like imputation of missing values will influence statistical results and so must be accounted for in the following analyses or interpretation thereof.

1.3 Known issues with the data and your planned solutions

In this section we discuss the possible issues that are know for this project or the big challenges and we manage those risk in order to provide a better solution.

1.3.1 520 independent variables

Due to the nature of wifi-printing we have about 520 independent variables to analyze, this is huge number for processing, that for a normal PC will take a lot of time for processing.

In order to simplify the analysis, and reduce the test-time, we proceed to use different technique or tools, that are listed below.

1.3.1.1 Delete the zero variance columns.

We use a function to delete all the independent variables that has zero variance, this help reducing the number of variables for the analysis.

This process is aply for the training data set as well as the validation, in order to match.

1.3.1.2 PCA

1.3.1.2.1 Definition PCA

The primary purpose of PCA is not as a ways of feature removal. PCA can reduce dimensionality but it wont reduce the number of features / variables in your data. What this means is that you might discover that you can explain 99% of variance in your 1000 feature dataset by just using 3 principal components but you still need those 1000 features to construct those 3 principal components, this also means that in the case of predicting on future data you still need those same 1000 features on your new observations to construct the corresponding principal components.

1.3.1.2.2 Procedure

1. Standardize the data (Center and scale).
2. Calculate the Eigenvectors and Eigenvalues from the covariance matrix or correlation matrix (One could also use Singular Vector Decomposition).
3. Sort the Eigenvalues in descending order and choose the K largest Eigenvectors (Where K is the desired number of dimensions of the new feature subspace $k \leq d$).
4. Construct the projection matrix W from the selected K Eigenvectors.
5. Transform the original dataset X via W to obtain a K-dimensional feature subspace Y.

1.3.2 Depended variables

As it's show in the table 1, to determine the localization of the person exist at least 10 variables to take into a count.

For the purpose of this training, we limit our model to determine to determine the location of the person of the building and the floor in which is located.

2. Selection of the models

2.1 Selection of the models

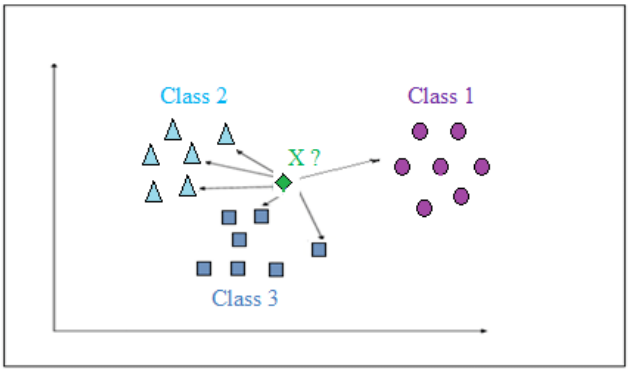
For the present project, 3 models have been selected. The selection of the models are base on the principal characteristic that the problem we want to resolve is a classification problem. Once the evaluation of the models are finish, their characteristics has been compare in order that select the model that has a better fit for the present problem.

2.2 k-nearest neighbors algorithm (k-NN)

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regrestion. In both cases, the input consists of the k closest training examples in the feature space.

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

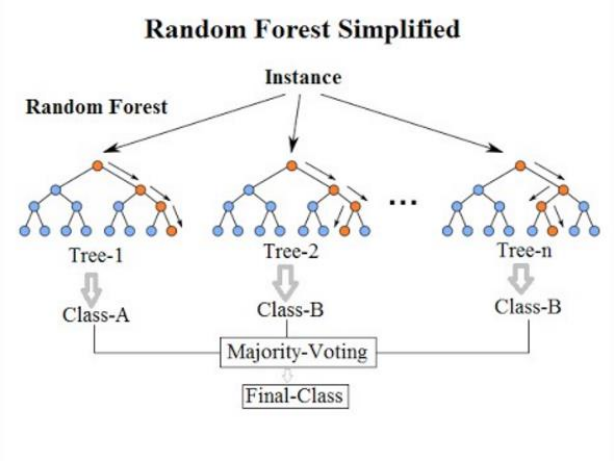
Feature extraction and dimension reduction can be combined in one step using principal component analysis (PCA), linear discriminant analysis (LDA), or canonical correlation analysis (CCA) techniques as a pre-processing step, followed by clustering by k-NN on feature vectors in reduced-dimension space. In machine learning this process is also called low-dimensional embedding.



2.3 Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

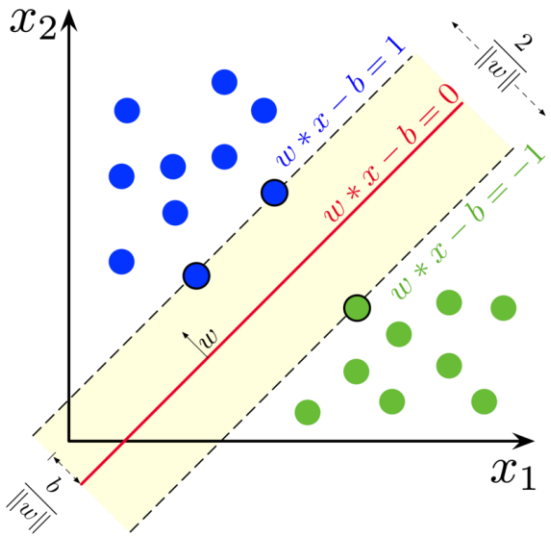
Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way. The following technique was described in Bierman's original paper and is implemented in the R package random Forest



2.4 Support-vector machine

In machine learning, support-vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.



3. Recommendation of the algorithm that has been use

The main goal of this project is IOT Analytics to evaluate if develop a solution that enables users to navigate in complex indoors spaces using WIFI fingerprinting is feasible.

In this case we propose to be able to predict where is located a person using the information of wifi location. For this purpose the solution was to predict for now the building and the floor in which is located the person.

For that we choose 4 algorithms of machine learning to evaluate which can give us the best performance to be able to predict. The algos that has been selected are:

- Support-Vector Machine
- Random Forest
- C50
- KNN

Base on the results that has been presented in the other documents and also attached in the present document as reference, we decided that the best performance algo is KNN.

This also was decided due to give us the best results in terms of test time. About 10 times less than Random forest that give the best results of kappa and accuracy. Actually, those results are 2% difference but still be on the 90%.

In the following table are shown the results of accuracy and kappa as reference.

Model	**Accuracy**	**Kappa**
RF	0.9545096	0.9503639
knn	0.9312672	0.9250274
C50	0.8956562	0.8861759
SVM	0.9466791	0.9418449

In terms of test time, we can see that KNN are shown a very good performance as mentioned before. This is very important, due to now a days what we should focus is more on the speed to have an idea where is located the person, and be ~93% are good results.

Algo	**user**	**system**	**elapsed**
RF	12948.79	62.96	13020.36
KNN	1516.97	11.89	1531.29
C50	2711.25	9.33	2721.97
SVM	5689.28	43.93	5738.09

4. Recommendations for future projects

In the following section based on our own research on indoor locating, there are some recommendations how the results might be improved.

4.1 Improve the tuning of the models

The first recommendation for improving the results is to have more time doing the tunning of the selected models. This can improve the results for the performance. In the present project we used different combination of tunning to improve the results. But with more time, exploring the combination of this value can be very beneficial.

4.2 Find more complex models

For this project the models that has been selected are the basic models as starting point. But there are locking for more complex algorithms can give better results.

For example the selected algorithm is KNN, but in the article "Enhanced Weighted K-Nearest Neighbor Algorithm for Indoor Wi-Fi Positioning Systems " is show how EWKNN has much better results.

4.3 Improving the pre-processing of the data

For this we used a reduction models in order to delete the independent variables that have zero variance and also the PCA it's described below to reducing the dimensionality and have faster results.

But for future projects more time can be expend looking for alternatives and compare which is the best way to proceed.

5. Results

5.1 Dimensionality Reductions

For this project, we have 520 independent variables, so this makes a complex analisys, and also cause a lot of test time. For this purpose, we use Principal Component Analysis (PCA). Actually, the primary purpose of PCA is not as a ways of feature removal. PCA can reduce dimensionality but it won't reduce the number of features / variables in your data. What this means is that you might discover that you can explain 99% of variance in your 1000 feature dataset by just using 3 principal components but you still need those 1000 features to construct those 3 principal components, this also means that in the case of predicting on future data you still need those same 1000 features on your new observations to construct the corresponding principal components.

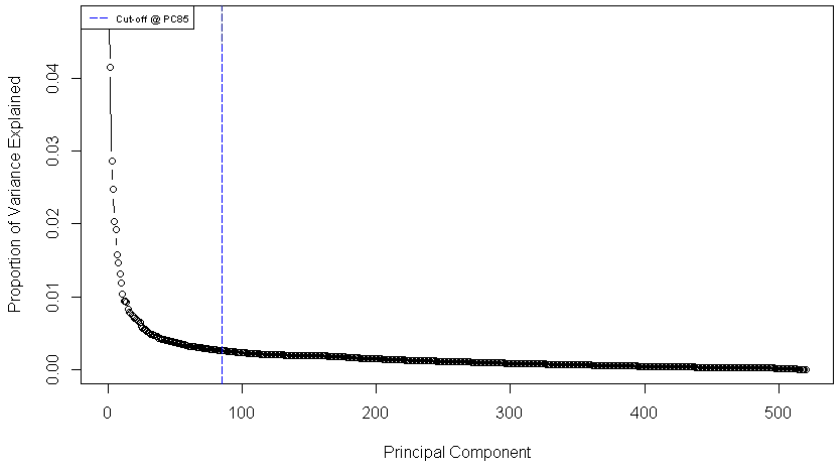
5.1.1 Results

After applying the PCA, we conclude that the number of variables that we require are 85, which give us 60%

```
> sum(prop_varex[1:Num_of_values]) #
[1] 0.6089494
```

5.1.2 Plotting PCA

In the following plot, is show the proportion of variance, in this case we should up to 85.



5.2 SVM

In machine learning, support-vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

5.2.1 Parameters

- 15791 samples
- 100 predictor
- 13 classes: '0 0', '0 1', '0 2', '0 3', '1 0', '1 1', '1 2', '1 3', '2 0', '2 1', '2 2', '2 3', '2 4'
- Pre-processing: centered (100), scaled (100)
- Resampling: Cross-Validated (10 fold)

5.2.2 Results

Resampling results across tuning parameters:

C	Accuracy	Kappa
0.25	0.8919043	0.8820412
0.50	0.9124817	0.9044980
1.00	0.9269187	0.9202537
2.00	0.9333158	0.9272575
4.00	0.9385088	0.9329298
8.00	0.9421190	0.9368704
16.00	0.9447175	0.9397052
32.00	0.9466791	0.9418449
64.00	0.9456031	0.9406706
128.00	0.9452239	0.9402552

5.2.3 Conclusion

Tuning parameter 'sigma' was held constant at a value of 0.01519876. Accuracy was used to select the o ptimal model using the largest value.
The final values used for the model were sigma = 0.01519876 and C = 32.

5.3 Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

5.3.1 Parameters

- 15791 samples
- 100 predictor
- 13 classes: '0 0', '0 1', '0 2', '0 3', '1 0', '1 1', '1 2', '1 3', '2 0', '2 1', '2 2', '2 3', '2 4'
- Pre-processing: centered (100), scaled (100)
- Resampling: Cross-Validated (10 fold, repeated 3 times)

5.3.2 Results

Resampling results across tuning parameters:

mtry	Accuracy	Kappa
4	0.9532638	0.9490024
5	0.9540666	0.9498794
6	0.9539182	0.9497173
8	0.9545096	0.9503639
10	0.9538129	0.9496036

5.3.3 Conclusion

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 8.

5.4 k-Nearest Neighbors

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regrestion. In both cases, the input consists of the k closest training examples in the feature space.

5.4.1 Parameters

- 15791 samples
- 100 predictor
- 13 classes: '0 0', '0 1', '0 2', '0 3', '1 0', '1 1', '1 2', '1 3', '2 0', '2 1', '2 2', '2 3', '2 4'
- Pre-processing: centered (100), scaled (100)
- Resampling: Cross-Validated (10 fold, repeated 3 times)

5.4.2 Results

Resampling results across tuning parameters:

k	Accuracy	Kappa
5	0.9312672	0.9250274
7	0.9256106	0.9188583
9	0.9194049	0.9120872
11	0.9140647	0.9062594
13	0.9093572	0.9011250
15	0.9067605	0.8982884
17	0.9010400	0.8920474
19	0.8972617	0.8879243
21	0.8951920	0.8856646
23	0.8925535	0.8827907

5.4.3 Conclusion

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was k = 5.

5.5 c50model

5.5.1 Parameters

- 15791 samples
- 100 predictor
- 13 classes: '0 0', '0 1', '0 2', '0 3', '1 0', '1 1', '1 2', '1 3', '2 0', '2 1', '2 2', '2 3', '2 4'
- Pre-processing: centered (100), scaled (100)
- Resampling: Cross-Validated (10 fold, repeated 3 times)

5.5.2 Results

Resampling results across tuning parameters:

model	winnow	Accuracy	Kappa
rules	FALSE	0.8951495	0.8856298
rules	TRUE	0.8956562	0.8861759
tree	FALSE	0.8931240	0.8834259
tree	TRUE	0.8932710	0.8835787

5.5.3 Conclusion

- Tuning parameter 'trials' was held constant at a value of 1
- Accuracy was used to select the optimal model using the largest value.
- The final values used for the model were trials = 1, model = rules and winnow = TRUE.

5.6 Test Time

The teatime is a critical factor in machine learning, this is due to amount of data. For instance if we have to an algorimt that has better performance, is not always the case in which this has beleted.

Algo	user	system	elapsed
RF	12948.79	62.96	13020.36
KNN	1516.97	11.89	1531.29
C50	2711.25	9.33	2721.97
SVM	5689.28	43.93	5738.09

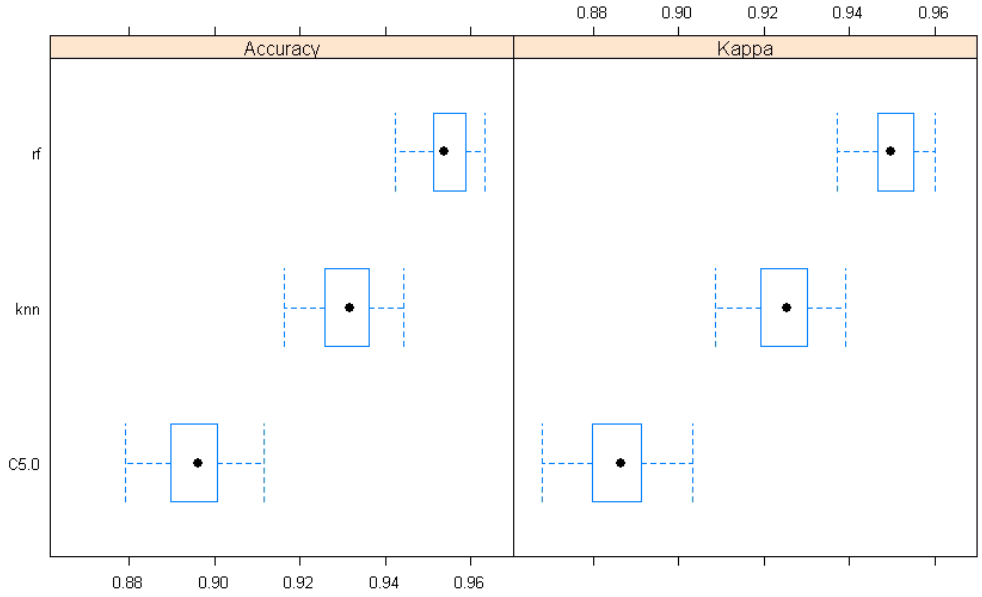
5.7 Comparing in terms of Resampling

Resampling methods are an indispensable tool in modern statistics. They involve repeatedly drawing samples from a training set and refitting a model of interest on each sample in order to obtain additional information about the fitted model. For example, in order to estimate the variability of a linear regression fit, we can repeatedly draw different samples from the training data, fit a linear regression to each new sample, and then examine the extent to which the resulting fits differ.

	Algo	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
Accuracy	C5.0	0.8790374	0.8897513	0.8961036	0.8956562	0.9004912	0.9113924	0
	rf	0.9423686	0.9513316	0.9538120	0.9545096	0.9586893	0.9632679	0
	knn	0.9162968	0.9260724	0.9316455	0.9312672	0.9359177	0.9442685	0
Kappa	C5.0	0.8680433	0.8797410	0.8866706	0.8861759	0.8913886	0.9033109	0
	rf	0.9371195	0.9468825	0.9496059	0.9503639	0.9549294	0.9599178	0
	knn	0.9086731	0.9193574	0.9254393	0.9250274	0.9300704	0.9391997	0

5.7.1 Plot

In this section we plot the different accuracy and kappa of the models that has been evaluate it. Having a visual representation is very useful when there is a need to explain which model is having the best performance.



5.8 Confusion Matrix

The confusion matrix calculates a cross-tabulation of observed and predicted classes with associated statistics. In this particular case we want to show the relationship between the different algorisms that has been used.

5.8.1 Accuracy

Accuracy is the percentage of correctly classifies instances out of all instances. It is more useful on a binary classification than multi-class classification problems because it can be less clear exactly how the accuracy breaks down across those classes.

	c5.0	**rf**	**knn**
c5.0		-0.05885	-0.05885
rf	< 2.2e-16		0.02324
knn	< 2.2e-16	< 2.2e-16	

5.8.2 Kappa

Kappa or Cohen's Kappa is like classification accuracy, except that it is normalized at the baseline of random chance on your dataset. It is a more useful measure to use on problems that have an imbalance

in the classes (e.g. 70-30 split for classes 0 and 1 and you can achieve 70% accuracy by predicting all instances are for class 0).

	c5.0	rf	knn
c5.0		-0.06419	-0.03885
rf	< 2.2e-16		0.02534
knn	< 2.2e-16	< 2.2e-16	

6. Model Selection and Analysis

In this section, we analyze which is the best model that performance for the wifi localization. It's important to mention that all these models the data has been preprocess using PCA to reduce the test time.

6.1 Model Selection

6.1.1 Selection based on performance

Base on the performance indicator, more specify in accuracy and kappa the Random Forest present the most accurate model base on the accuracy and kappa. In second place we can review that knn and the different is about 2%.

Model	Accuracy	Kappa
RF	0.9545096	0.9503639
knn	0.9312672	0.9250274
C50	0.8956562	0.8861759
SVM	0.9466791	0.9418449

6.1.2Selection based on Test time

In data science another critical indicator in order to choose the model, besides their performance it's the test time that require to build the model. For example, in these cases we notice that the KNN is running 8 time faster of the random forest, but when the performance is compare, there is not huge difference it's about 2%.

Algo	user	system	elapsed
RF	12948.79	62.96	13020.36
KNN	1516.97	11.89	1531.29
C50	2711.25	9.33	2721.97
SVM	5689.28	43.93	5738.09

6.1.3 Conclusion of the model selected

Base on the test time and the performance we conclude that the best model is the SVM, due to it took less than the half of the time than RF to complete and the performance is 2% less, and actually is ~93. So still a very good model.

6.2 k-Nearest Neighbors

In pattern recognition, the k-nearest neighbors' algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space.

6.2.1 Confusion Matrix

Calculates a cross-tabulation of observed and predicted classes with associated statistics.

A confusion matrix is a summary of prediction results on a classification problem.

The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix.

The confusion matrix shows the ways in which your classification model is confused when it makes predictions.

It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

		Reference												
		0_0	0_1	0_2	0_3	1_0	1_1	1_2	1_3	2_0	2_1	2_2	2_3	2_4
Prediction	0_0	261	15	3	0	0	0	0	0	0	0	0	0	0
	0_1	19	365	17	5	0	0	0	0	0	0	0	0	0
	0_2	3	9	339	29	0	0	1	0	0	0	0	0	0
	0_3	0	1	43	334	0	0	1	0	0	0	0	0	0
	1_0	0	0	0	0	342	14	1	0	0	0	0	0	0
	1_1	0	0	0	0	6	386	5	0	0	0	0	0	0
	1_2	0	0	0	0	1	5	347	8	0	0	0	0	0
	1_3	1	1	0	0	0	1	14	234	3	0	0	0	0
	2_0	0	0	0	0	0	0	0	5	482	7	1	1	0
	2_1	0	0	0	0	0	0	1	0	5	547	18	5	1
	2_2	0	0	0	1	0	0	0	0	1	9	368	15	0
	2_3	0	0	0	0	0	0	0	1	0	5	20	665	12
	2_4	0	0	0	0	0	0	0	0	0	0	0	1	272

6.2.2 Overall statistics

- Accuracy : 0.9401
- 95% CI : (0.9333, 0.9463)
- No Information Rate : 0.1307
- P-Value [Acc > NIR] : < 2.2e-16

	Class : 0_0	Class : 0_1	Class : 0_2	Class : 0_3	Class : 1_0	Class : 1_1	Class : 1_2	Class : 1_3	Class : 2_0	Class : 2_1	Class : 2_2	Class : 2_3	Class : 2_4
Sensitivity	0,92	0,93	0,84	0,91	0,98	0,95	0,94	0,94	0,98	0,96	0,90	0,97	0,95
Specificity	1,00	0,99	0,99	0,99	1,00	1,00	1,00	1,00	1,00	0,99	0,99	0,99	1,00
Pos Pred Value	0,94	0,90	0,89	0,88	0,96	0,97	0,96	0,92	0,97	0,95	0,93	0,95	1,00
Neg Pred Value	1,00	0,99	0,99	0,99	1,00	1,00	1,00	1,00	1,00	1,00	0,99	1,00	1,00
Prevalence	0,05	0,07	0,08	0,07	0,07	0,08	0,07	0,05	0,09	0,11	0,08	0,13	0,05
Detection Rate	0,05	0,07	0,06	0,06	0,07	0,07	0,07	0,04	0,09	0,10	0,07	0,13	0,05
Detection Prevalence	0,05	0,08	0,07	0,07	0,07	0,08	0,07	0,05	0,09	0,11	0,07	0,13	0,05
Balanced Accuracy	0,96	0,96	0,92	0,95	0,99	0,97	0,97	0,97	0,99	0,98	0,95	0,98	0,98