DATA SCIENCE AND PYTHON

CAPSTONE PROJECT

TECHNIQUES FOR WIFI LOCATIONING

BY AMARENDRA GHANEKAR

APRIL 30, 2018

1 BIGDATA STREET, OTTAWA, ON
CANADA

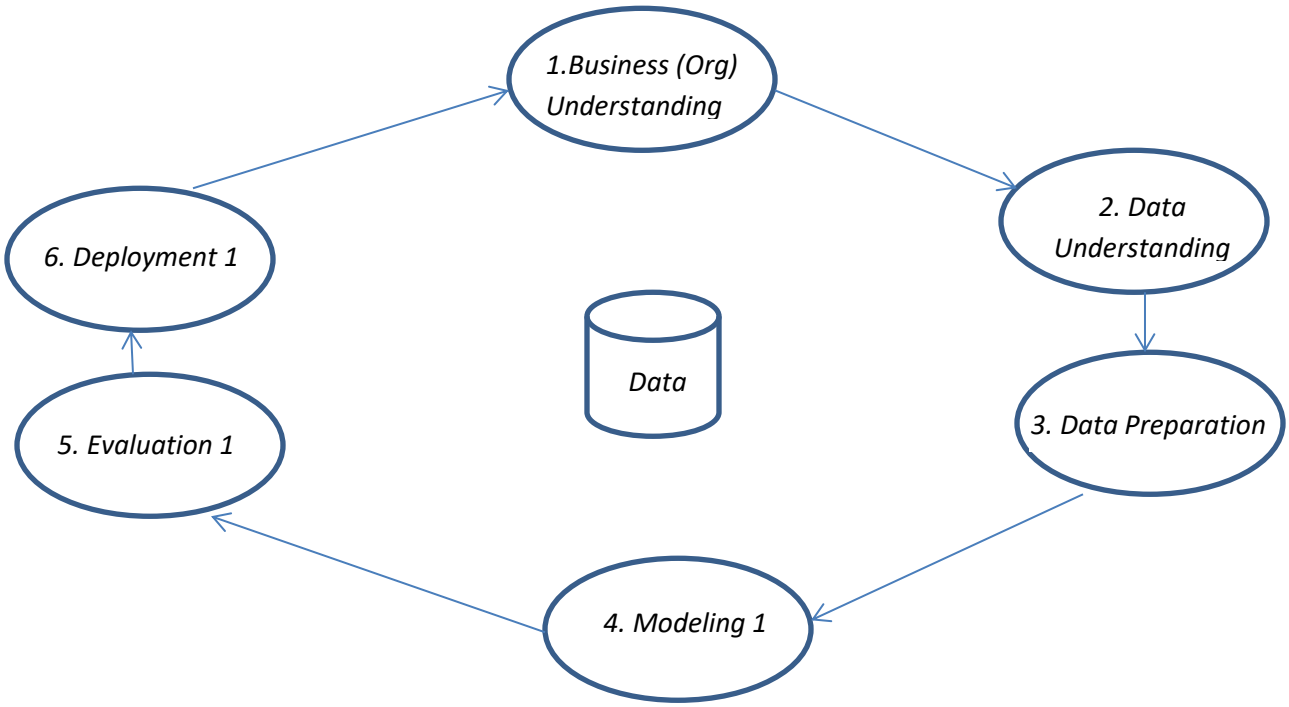
Table of Contents

Project Background

The client wants to develop a system to be deployed on large industrial campuses, in shopping malls, et cetera to help people to navigate a complex, unfamiliar interior space without getting lost. The task given was to do feasibility of using "Wi-Fi" fingerprinting" to determine a person's location in indoor spaces. Wi-Fi fingerprinting uses the signals from multiple Wi-Fi hotspots within the building to determine location, analogously to how GPS uses satellite signals.

The source data given was a large database of Wi-Fi fingerprints for a multi-building industrial campus with a location (building, floor, and location ID) associated with each fingerprint. My goal was to evaluate multiple machine learning models to see which produces the best result.

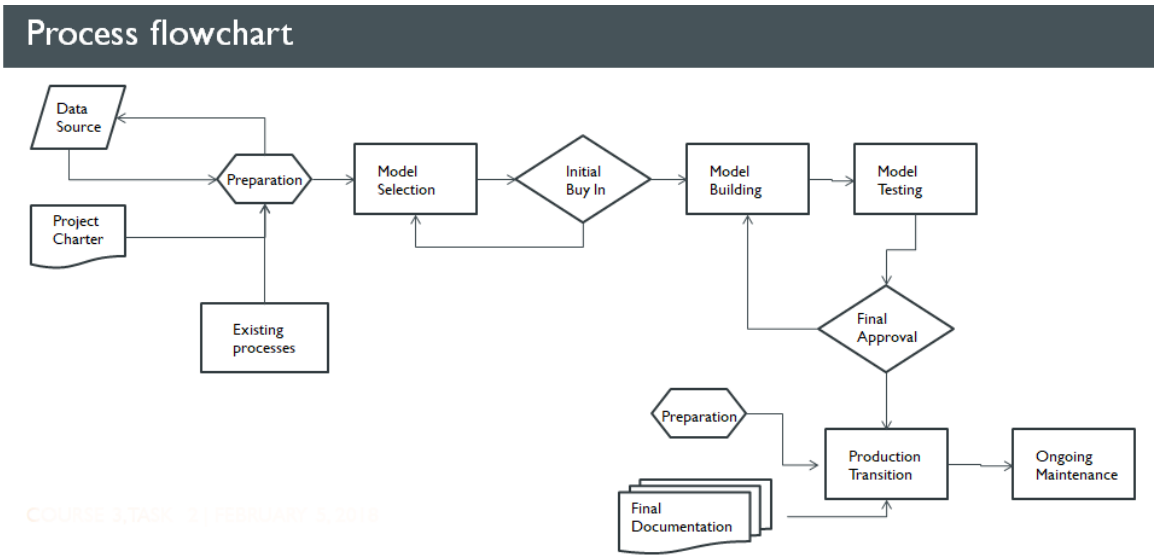
While working on this project, I followed **CRISP-DM (**CRoss-Industry Standard Process for Data Mining), which consists of six steps or phases, as illustrated below:



CRISP-DM Conceptual Model.

(Reference: Data Mining for the Masses by Dr. Matthew North)

In this project, I followed the process flow as given in the diagram below:



Source Data and Data management

Following data was made available for this project:

Multi-Building Multi-Floor indoor localization database to test Indoor Positioning System that relies on WLAN/WiFi fingerprint.

Data Set Characteristics:	Multivariate	**Number of Instances:**	21048
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	529
Associated Tasks:	Classification	**Missing Values?**	N/A

The database covers three buildings 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 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).

Data Attributes Information:

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. Measures were taken in three different buildings. Categorical integer values from 0 to 2.

Attribute 525 (SpaceID): Internal 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 (see below). Categorical integer values.

Attribute 528 (PhoneID): Android device identifier (see below). Categorical integer values.

Attribute 529 (Timestamp): UNIX Time when the capture was taken. Integer value.

UserID Anonymized user Height (cm)

0 USER0000 (Validation User) N/A

1 USER0001 170

2 USER0002 176

3 USER0003 172

4 USER0004 174

5 USER0005 184

6 USER0006 180

7 USER0007 160

8 USER0008 176

9 USER0009 177

10 USER0010 186

11 USER0011 176

12 USER0012 158

13 USER0013 174

14 USER0014 173

15 USER0015 174

16 USER0016 171

17 USER0017 166

18 USER0018 162

PhoneID Android Device Android Ver. UserID

0 Celkon A27 4.0.4(6577) 0

1 GT-I8160 2.3.6 8

2 GT-I8160 4.1.2 0

3 GT-I9100 4.0.4 5

4 GT-I9300 4.1.2 0

5 GT-I9505 4.2.2 0

6 GT-S5360 2.3.6 7

7 GT-S6500 2.3.6 14

8 Galaxy Nexus 4.2.2 10

9 Galaxy Nexus 4.3 0

10 HTC Desire HD 2.3.5 18

11 HTC One 4.1.2 15

12 HTC One 4.2.2 0

13 HTC Wildfire S 2.3.5 0,11

14 LT22i 4.0.4 0,1,9,16

15 LT22i 4.1.2 0

16 LT26i 4.0.4 3

17 M1005D 4.0.4 13

18 MT11i 2.3.4 4

19 Nexus 4 4.2.2 6

20 Nexus 4 4.3 0

21 Nexus S 4.1.2 0

22 Orange Monte Carlo 2.3.5 17

23 Transformer TF101 4.0.3 2

24 bq Curie 4.1.1 12

Data management process:

- Data processed and stored on Amazon Web Services (AWS) Cloud.
- AWS environments are continuously audited for various industry certifications
- AWS provides compliance with 20+ standards.
- Connectivity options that enable private, or dedicated, connections to be used as necessary
- Ongoing data acquisition, access and retention policies to be established and followed.

Data Analysis and preparation

Initial analysis of the data showed the distributions of records within the buildings and floors as per the table given below.

	Building ID 0	Building ID 1	Building ID 2	Total
Floor 0	1059	1368	1942	**4369**
Floor 1	1356	1484	2162	**5002**
Floor 2	1443	1396	1577	**4416**
Floor 3	1391	948	2709	**5048**
Floor 4	0	0	**1102**	**1102**
	5249	**5196**	**9492**	**19937**

I devised a single unique identifier for each location in the ample dataset. I combined building ID, floor and location to create the unique identifier, which is used for prediction. This unique identifier is of float type.

To avoid processing power challenges I selected building 2 floor 4 as the sample size to test.

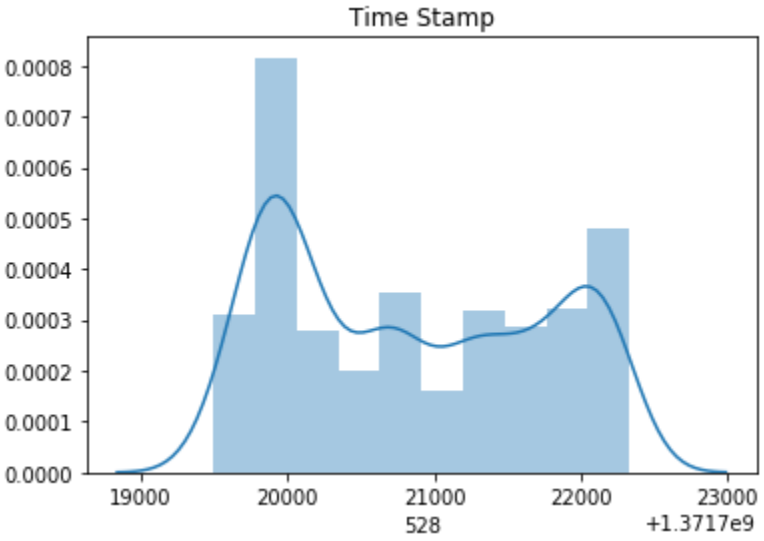
Understand the makeup of the data:

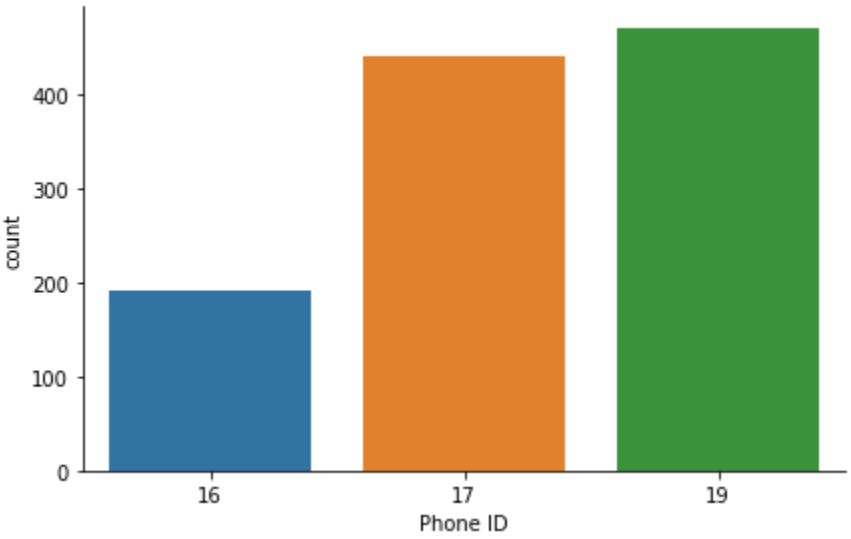
0	1	2	3	4	5	6	7	8	9	...	520	521	522	523	524	525	526	527	528	529	
count	1102	1102	1102	1102	1102	1102	1102	1102	1102	1102	...	1102	1.10E+03	1102	1102	1102	1102	1102	1102	1.10E+03	1102
mean	100	100	100	100	98.2577	100	100	100	100	100	...	-7357.5	4.86E+06	4	2	138.404	1.98185	8.27223	17.6788	1.37E+09	42138.4
std	0	0	0	0	18.215	0	0	0	0	0	...	30.8889	2.15E+01	0	0	35.094	0.13355	3.99799	1.19197	8.90E+02	35.094
min	100	100	100	100	-92	100	100	100	100	100	...	-7414	4.86E+06	4	2	101	1	3	16	1.37E+09	42101
25%	100	100	100	100	100	100	100	100	100	100	...	-7381.8	4.86E+06	4	2	114	2	6	17	1.37E+09	42114
50%	100	100	100	100	100	100	100	100	100	100	...	-7359.3	4.86E+06	4	2	129	2	6	17	1.37E+09	42129
75%	100	100	100	100	100	100	100	100	100	100	...	-7328	4.86E+06	4	2	141	2	13	19	1.37E+09	42141
max	100	100	100	100	100	100	100	100	100	100	...	-7309.5	4.86E+06	4	2	220	2	13	19	1.37E+09	42220

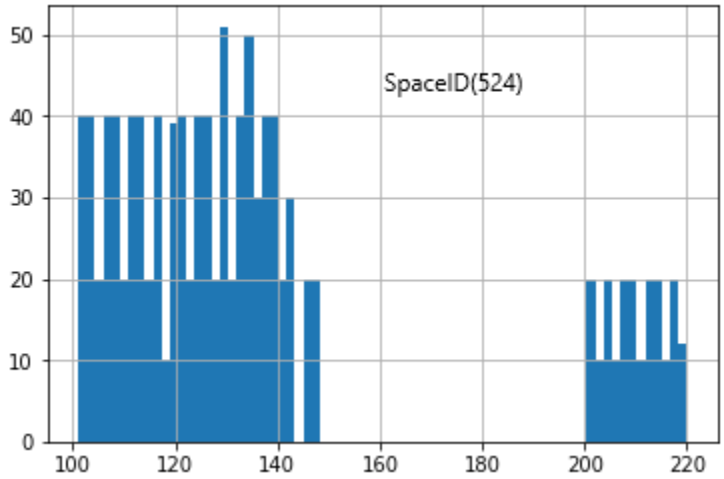
Data was checked for missing values, and, correlation and covariation was used to understand the importance of features in modeling.

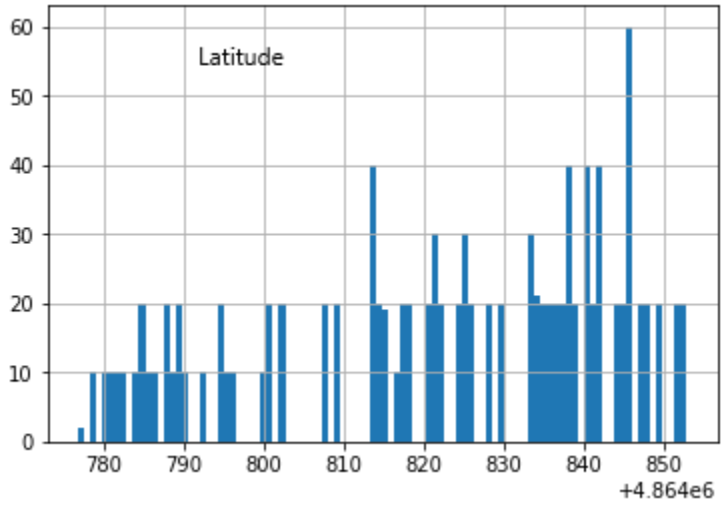
- **Correlation:** is a measure that determines the degree to which two variables' movements are associated
- **Covariance:** is a measure of how changes in one variable are associated with changes in a second variable. Specifically, covariance measures the degree to which two variables are linearly associated. A positive covariance means that asset returns move together, while a negative covariance means returns move inversely.

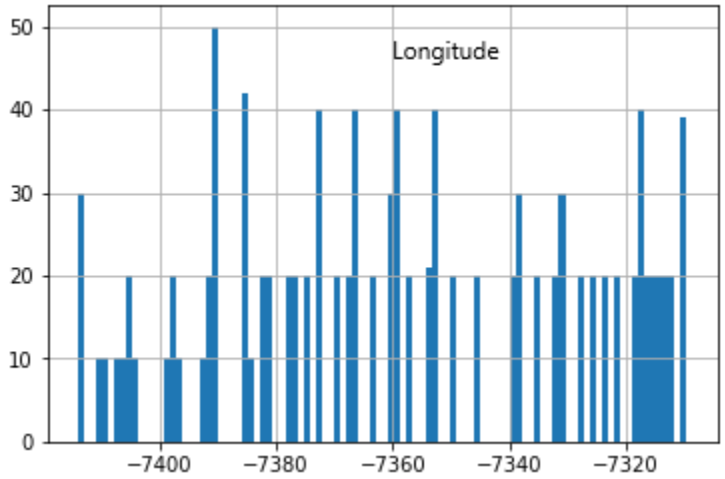
Various plots produced during data analysis phase are given below:

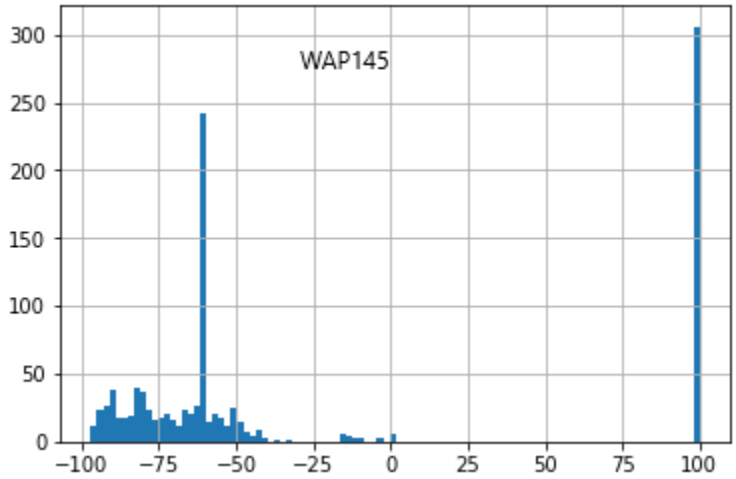





SpaceID(524)
0
10
20
30
40
50
100
120
140
160
180
200
220


Latitude
0
10
20
30
40
50
60
780
790
800
810
820
830
840
850
+4.864e6


Longitude
50
40
30
20
10
0
−7400
−7380
−7360
−7340
−7320


WAP145
300
250
200
150
100
50
0
−100
−75
−50
−25
0
25
50
75
100

Feature Engineering

I used RandomForestClassifier () to rank features based on importance. Listed below are top 20 features ranked by using RF:

524	0.055592
520	0.043824
521	0.035889
144	0.029727
137	0.028182
62	0.028067
143	0.023884
248	0.023694
528	0.022777
63	0.022448
480	0.022418
86	0.022114
84	0.021980
397	0.021516
516	0.020698
82	0.020410
138	0.020336
83	0.019467
10	0.018205
495	0.017429

There are multiple features with a constant value of 100, which do not add any value in predicting process.

Classification, model selection and model tuning:

This is a classification problem, and I tested following classification models:

1. **k-nearest neighbors** - KNeighborsClassifier() - AmarendraGhanekarC5T3NN.ipynb
2. **Random Forest** – RandomForestClassifier ()- AmarendraGhanekarC5T3RF.ipynb
3. **Support Vector Machines** - svm.SVC() - AmarendraGhanekarC5T3SVM.ipynb
4. **Decision Tree** – tree.DecisionTreeClassifier() - AmarendraGhanekarC5T3DTREEipynb
5. **Extra Trees** - ExtraTreesClassifier() - AmarendraGhanekarC5T3ETREE.ipynb

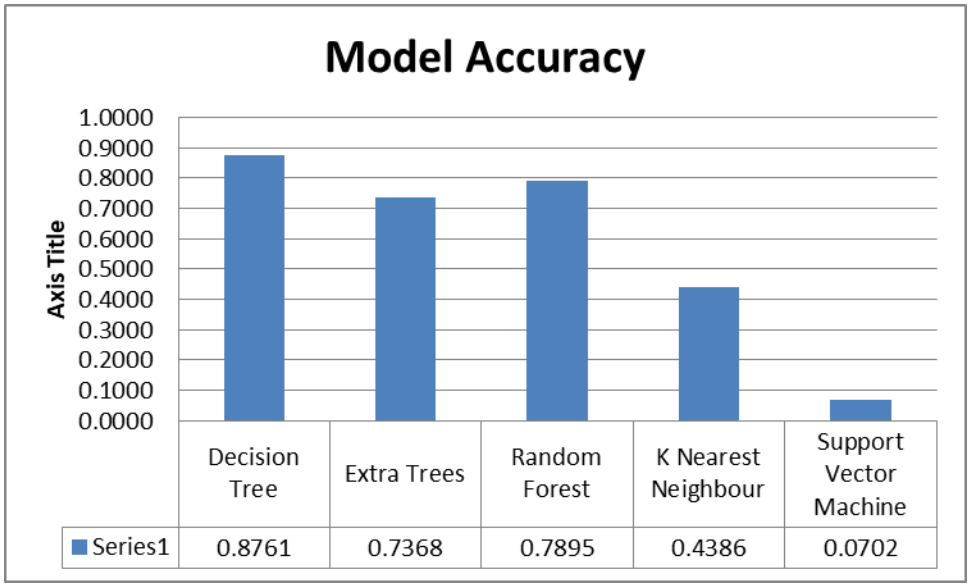
For each model following scoring classifications were measured and compared to arrive at the best model suitable for the data:

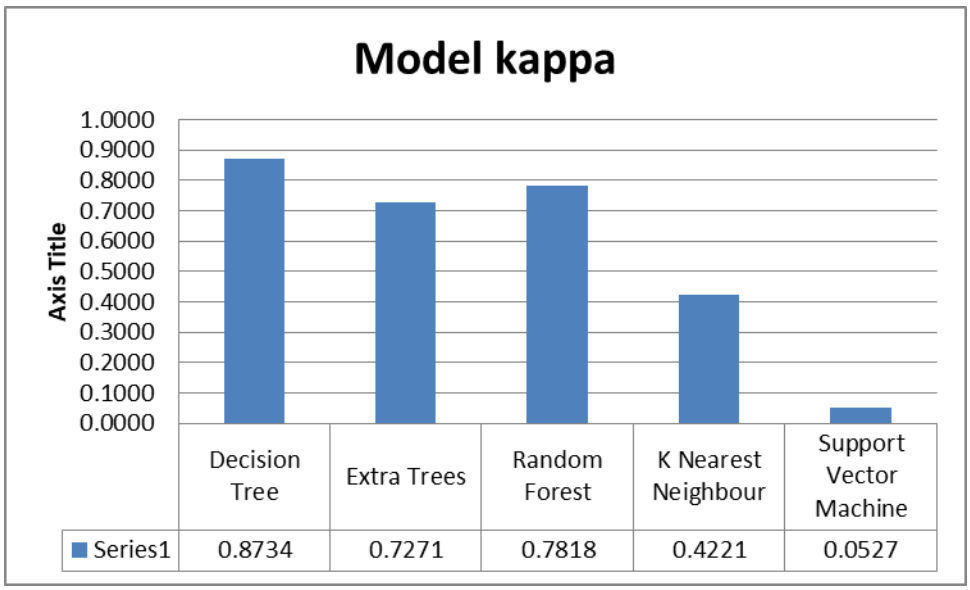
1. **Accuracy:** This function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y_true.
2. **Kappa:** This function computes Cohen's kappa, a score that expresses the level of agreement between two annotators on a classification problem.
3. **Precision:** The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative. The best value is 1 and the worst value is 0.
4. **Recall:** The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples. The best value is 1 and the worst value is 0.
5. **F1-score:** The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is: F1 = 2 * (precision * recall) / (precision + recall).
6. **Cross Validation Score:** This function returns 'test score', 'fit_time', and 'score_time'.

Comparison table of scoring classification for all models is given below:

#	Model	Cross Validation Score			Accuracy	Kappa	Precision	Recall	f1-score
		Score array for test scores on each cv split	Time for fitting the estimator on the train set for each cv split	Time for scoring the estimator on the test set for each cv split.					
1	Decision Tree	0.7687	0.8230	0.8462	0.8761	0.8734	0.9000	0.8800	0.8700
2	Extra Trees	0.5309	0.6481	0.8235	0.7368	0.7271	0.7200	0.7400	0.7200
3	Random Forest	0.6750	0.9091	0.9429	0.7895	0.7818	0.7700	0.7900	0.7700
4	K Nearest Neighbour	0.2561	0.4444	0.6471	0.4386	0.4221	0.4100	0.4400	0.4000
5	Support Vectoe Matrix	0.1220	0.1481	0.1176	0.0702	0.0527	0.1100	0.0700	0.0700

Graphical comparison of accuracy and kappa for all models tested is given below:

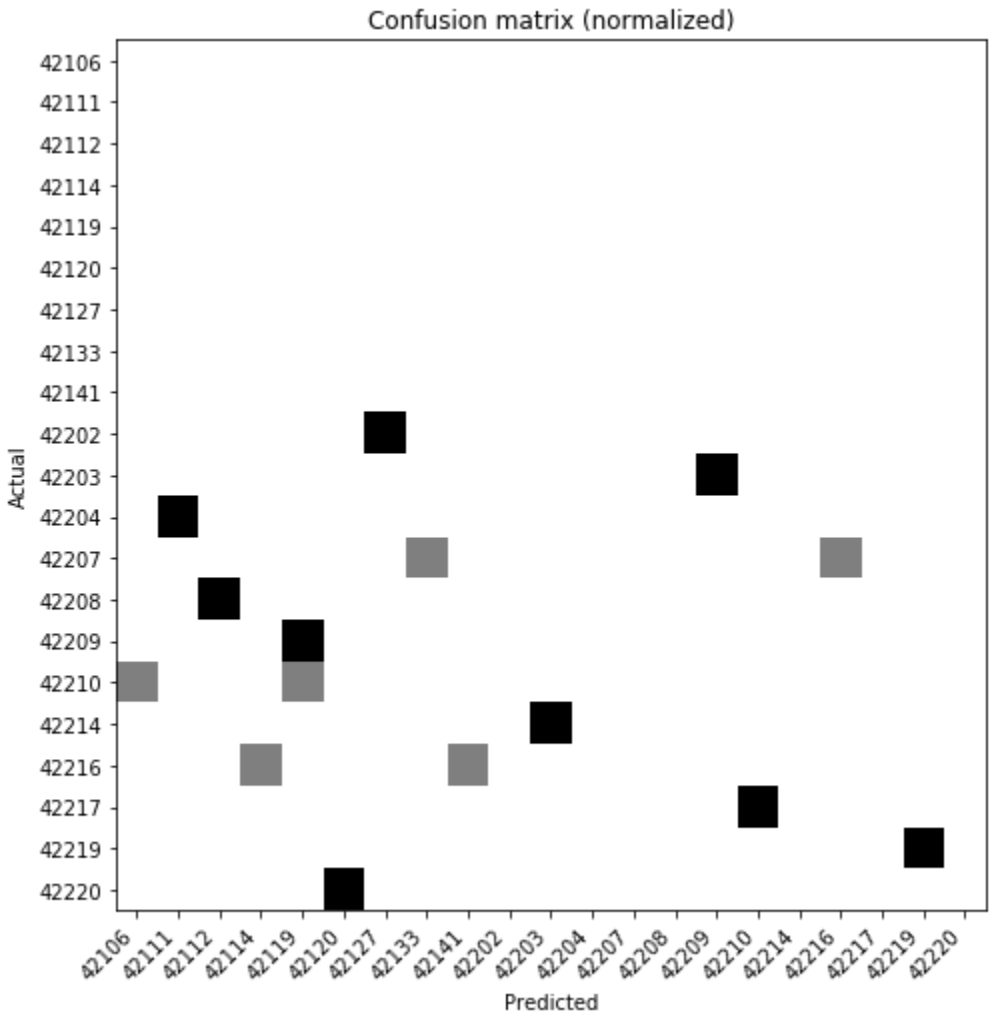




Decision Tree model was selected and fine-tuned to predict the unique identifier for the given data. This model provided following performance:

1. Accuracy = 0.8761
2. Kappa= 0.8734
3. Precision =0.9000
4. Recall = 0.8800
5. f1-score = 0.8700
6. Test score = 0.76865672
7. fit_time = 0.82300885
8. score_time = 0.84615385

Confusion matrix plot for the Decision Tree is given below:



Lessons Learned and Recommendations

1. The Decision Tree model can be deployed to do automatic user localization in real life situation, although periodic cross evaluation and data analysis is recommended.
2. In this project, classification models were used to predict location using unique identifier comprised of building floor and location. Further analysis is recommended to use regression models to predict longitude and latitude.
3. Decision Tree provided best overall results. SVM model was slowest and with large dataset it may not be effective.
4. The size of the data base is too large for the hardware and software environment available for this project. Python is unable to read the data provided due to memory error. A subset of the data was used to build and test models. More powerful environment and further efforts on tuning the parameters is recommended in order to analyze the complete data available.

References:

1. Joaquín Torres-Sospedra, Raúl Montoliu, Adolfo Martínez-Usó, Tomar J. Arnau, Joan P. Avariento, Mauri Benedito-Bordonau, Joaquín Huerta. UJIIndoorLoc: A New Multi-building and Multi-floor Database for WLAN Fingerprint-based Indoor Localization Problems. In Proceedings of the Fifth International Conference on Indoor Positioning and Indoor Navigation, 2014.

2. UCI Machine Learning Repository - Center for Machine Learning and Intelligent Systems.

3. Data Mining for the Masses by Dr. Matthew North.

4. Data Mining: Practical Machine Learning Tools and Techniques – Ian H. Witten, Eibe Frank, Mark A. Hall and Christopher J. Pal.

5. scikit-learn- Machine Learning in Python - http://scikit-learn.org/stable/