MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

BHOPAL (M.P.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MAJOR PROJECT

ON

Indoor Positioning System Using Wifi

SUBMITTED IN PARTIAL FULFILLMENT FOR THE DEGREE OF BACHELOR OF TECHNOLOGY

SUBMITTED BY:

GOUTHAM KALLEMPUDI(141112050)

RITWIJ SHUKLA (141112256)

BANWARI LAL SAINI(141112275)

SANA ISLAM DOWEL(141112094)

UNDER THE GUIDANCE OF:

DR. MANISH PANDEY

SESSION 2017-2018

MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

BHOPAL(M.P)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that **GOUTHAM KALLEMPUDI**, **RITWIJ SHUKLA**, **BANWARI LAL SAINI** and **SANA ISLAM DOWEL** students of B.Tech Final Year (Computer Science & Engineering), have successfully completed their major project on "**INDOOR POSITIONING SYSTEM USING WIFI**" in partial fulfilment of their major project in Computer Science & Engineering.

DR.JYOTHI BARTHI (Project Coordinator)

DR.MANISH PANDEY (Project Guide)

MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY

BHOPAL (M.P)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We, hereby, declare that the following report which is being presented in the Major Project Documentation entitled "**INDOOR POSITIONING SYSTEM USING WIFI**" is the partial fulfilment of the requirements of the final year (8th semester) Major Project in the field of Computer Science and Engineering. It is an authentic documentation of our own original work carried out under the able guidance of Dr.Manish Pandey. The work has been carried out entirely at Maulana Azad National Institute of Technology, Bhopal. The following project and its report, in part or whole, has not been presented or submitted by us for any purpose in any other institute or organization.

We, hereby, declare that the facts mentioned above are true to the best of our knowledge. In case of any unlikely discrepancy that may possibly occur, we will be the ones to take responsibility.

GOUTHAM KALLEMPUDI (141112050)

RITWIJ SHUKLA (141112050)

BANWARI LAL SAINI (141112275)

SANA ISLAM DOWEL (141112094)

ACKNOWLEDGEMENT

With due respect, we express our deep sense of gratitude to our respected guide Dr. Manish Pandey, for his valuable help and guidance. We are very thankful for the encouragement that he has given us in completing this project successfully. His rigorous evaluation and constructive criticism was of great assistance.

It is imperative for us to mention the fact that this major project could not have been accomplished without the periodic suggestions and advice of our project co-ordinator Dr.Jyothi Barthi.

We are also grateful to our respected Director Dr. N. S Raghuvanshi for permitting us to utilize all the necessary facilities of the college.

Needless to mention is the additional help and support extended by our respected HOD, Dr. Meenu Chawla, in allowing us to use the departmental laboratories and other services.

We are also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind co-operation and help.

Last but certainly not the least, we would like to express our deep appreciation towards our family members and batch mates for providing the much needed support and encouragement.

CONTENTS

1. Introduction

Indoor Positioning System (IPS) aims at wirelessly locating objects or people inside buildings based on magnetic sensor network, or other source of data. The major consumer benefit of indoor positioning is the expansion of location-aware mobile computing indoors, such as augmented reality, store navigation, etc. As mobile devices become ubiquitous, contextual awareness for applications has become a priority for developers. Most applications currently rely on GPS, however, and function poorly indoors.

Due to the proliferation of both wireless local area networks (WLANs) and mobile devices, WiFi-based IPS has become a practical and valid approach for IPS, that does not require extra facility cost. However, Wifi-based position system as (WPS) accuracy depends on the number of positions that have been entered into the database.

2. Abstract

Indoor Positioning System aims at locating objects inside buildings wirelessly. Have huge benefit for indoor location-aware mobile application. To explore this immature system design, we choose UJIndoorLoc database as our data set, use PCA for feature selection, and build prediction models based on decision tree, gradient boosting, KNN and SVM, respectively.

Thus, with given roadmap of fingerprints as training set, we could generate a model using machine learning techniques, with which we would be able to predict the building number of unknown mobile device holder in a certain building.

3. Machine learning

Machine learning is a branch of computer science that relies on the implementation of artificial intelligence that enables the computers to learn and perform functionalities without explicitly programming them. It is broadly classified into 3 parts Supervised learning, Unsupervised learning and Reinforcement.

3.1 Supervised Learning

Supervised learning is the machine learning task of inferring a function from labelled training data. The training data consist of a set of training examples., In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal).

In supervised learning, both the input and the desired output is provided and this data is labelled for classification to provide a learning basis for future data processing.

3.2 Unsupervised Learning

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses. In unsupervised learning, no datasets are provided, instead the data is clustered into different classes.

3.3 Reinforcement Learning

Reinforcement Learning is another type of Machine Learning where it allows machines and software agents to automatically determine the ideal behaviour within a specific context, in order to maximize its performance. Simple reward feedback is required for the agent to learn its behaviour; this is known as the reinforcement signal.

In this project, we make use of the Supervised Learning by implementing Logistic Regression and SVM.

Supervised learning problems are again categorized into Regression and Classification.

Classification: A classification problem is when the output variable is a category, such as "red" or "blue" or "disease" and "no disease".
Example: Logistic Regression, SVM.

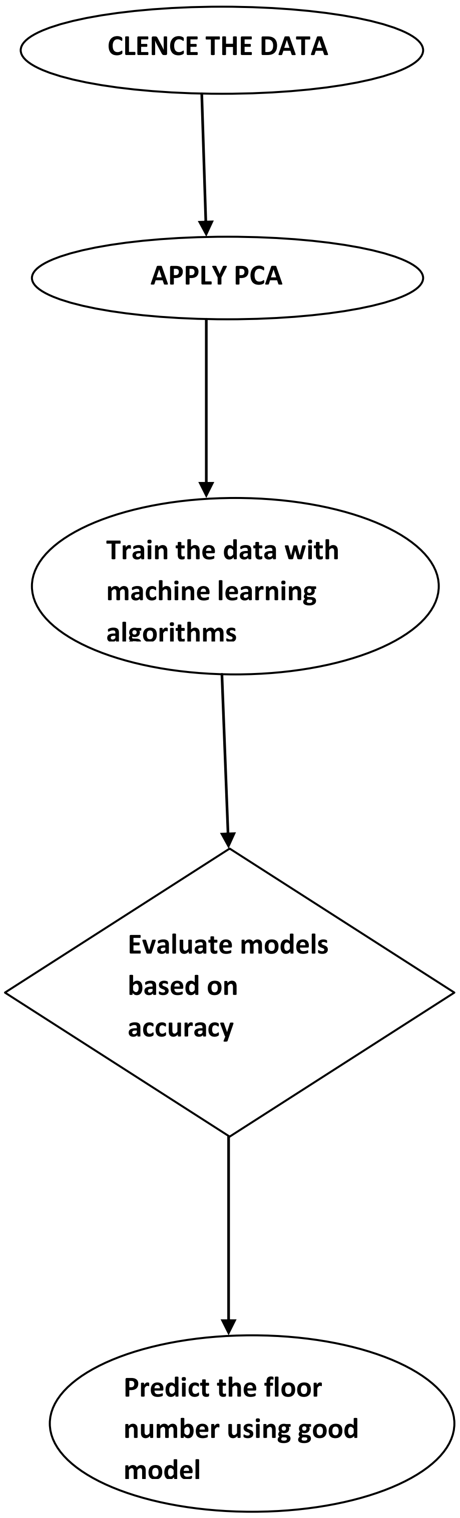
Regression: A regression problem is when the output variable is a real value, such as "dollars" or "weight".
Some common types of problems built on top of classification and regression include recommendation and time series prediction respectively.

Example: Linear Regression.

Some popular examples of supervised machine learning algorithms are:

- Linear regression for regression problems.
- Random forest for classification and regression problems.
- Support vector machines for classification problems.

4. FLOW DIAGRAM:


CLENCE THE DATA
APPLY PCA
Train the data with machine learning algorithms
Evaluate models based on accuracy
Predict the floor number using good model

5. SOFTWARE AND HARDWARE REQUIREMENTS

SOFTWARE:

The following soft-wares were used for this project:

- Operating System – Microsoft® Windows® 8 , Ubuntu or above.
- Jupyter NoteBook – To run codes of Indoor Positioning System using Wifi.
- Microsoft Excel- To take dataset as input and for pre-processing of dataset.

HARDWARE:

The following hardware configuration is required to run the various soft-wares for this project:

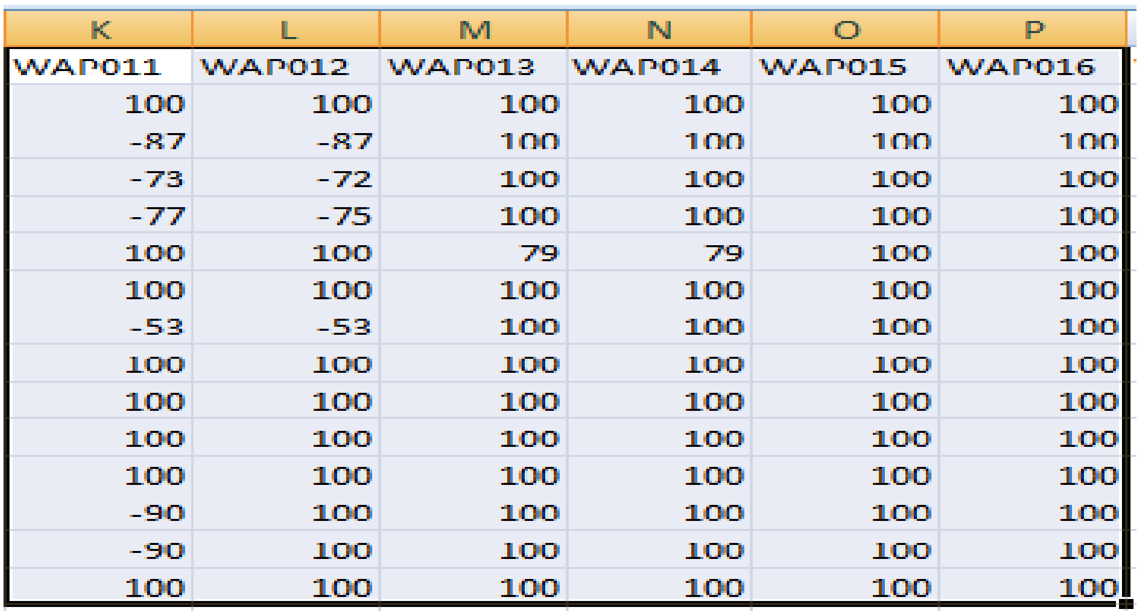
- Processor: Intel® Core™ i3 CPU
- Memory: 4GB RAM
- Storage required: Maximum of 100MB
- Graphics card: not needed

6. Dataset :

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 known as WiFi Fingerprinting).

The **UJIIndoorLoc** database covers three buildings of **Universitat JaumeI** 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).



K	L	M	N	O	P
WAP011	WAP012	WAP013	WAP014	WAP015	WAP016
100	100	100	100	100	100
-87	-87	100	100	100	100
-73	-72	100	100	100	100
-77	-75	100	100	100	100
100	100	79	79	100	100
100	100	100	100	100	100
-53	-53	100	100	100	100
100	100	100	100	100	100
100	100	100	100	100	100
100	100	100	100	100	100
100	100	100	100	100	100
-90	100	100	100	100	100
-90	100	100	100	100	100
100	100	100	100	100	100

6.1. Data

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 and floor) and Building ID are provided as the attributes to be predicted. .

6.2. Featureset :

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 value 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. They are 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 4.

Attribute 525 (SpaceID): Internal ID number to identify the Space (office, corridor and classroom) where the capture was taken they are categorical integer values.

Attribute 526 (RelativePosition): Relative position with respect to the Space (1 - Inside, 2 - Outside in Front of the door) they are categorical integer values.

Attribute 527 (UserID): User identifier (see below) they are categorical integer values.

Attribute 528 (PhoneID): Android device identifier (see below they are categorical integer values.

Attribute 529 (Timestamp): UNIX Time when the capture was taken they are integer values.

6.3. RSSI STRENGTH:

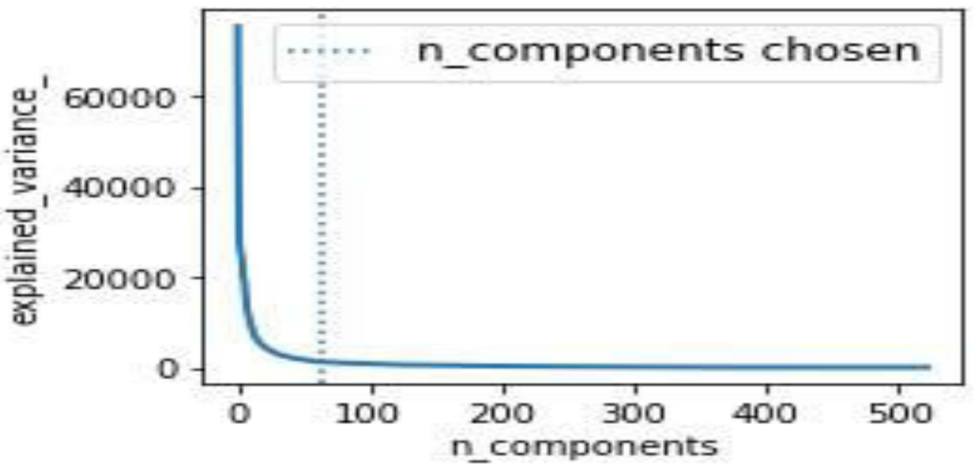
We use UJIndoorLoc database for this project. It consists of 520 RSSI fingerprints detected using 16 different phone models by 18 users from 4 to 5 different floors of 4 buildings. For each building, this dataset gives us thousands of RSSI samples generated at various locations inside the building.

RSSI value ranges between -104db (poor signal) to 0db (good signal) and 100 as no signal.

Thus, with given roadmap of fingerprints as training set, we could generate a model using machine learning techniques, with which we would be able to predict the floor number of unknown mobile device holder in a certain building

7. DIMENSIONALITY REDUCTION

However, the high dimensional feature space with redundant features would hurt computational efficiency. Therefore, we used Principal Component Analysis (PCA) to extract principal features. The energy levels (explained variance) of the principal components are shown in figure. We found that the top 64 principal components contain most of the energy, and for the components beyond 100, each of their energy levels is less than one. As we can see from figure, there is a trade-off between prediction accuracy and number of features required, which indicates the computation complexity, for the learning task. Depending on the characteristics of logistic Regression learning algorithm, we choose 64 top features for the suitable algorithms to compare and explore for the best prediction accuracy.



7.1. Summarizing the PCA approach

Listed below are the 6 general steps for performing a principal component analysis, which we will investigate in the following sections.

- Take the whole dataset consisting of d-dimensional samples ignoring the class labels.
- Compute the d-dimensional mean vector (i.e., the means for every dimension of the whole dataset).

$$\boldsymbol{m} = \frac{\boldsymbol{1}}{\boldsymbol{n}} * \sum_{\boldsymbol{1}}^{\boldsymbol{k}} \boldsymbol{x_k}$$

- Compute the scatter matrix (alternatively, the covariance matrix) of the whole data set.

$$\boldsymbol{s} = \sum_{\boldsymbol{1}}^{\boldsymbol{k}} (\boldsymbol{x_k} - \boldsymbol{1})(\boldsymbol{x_k} - \boldsymbol{1})^{\boldsymbol{T}}$$

- Compute eigenvectors (e1,e2,...,ed) and corresponding Eigen values (λ1,λ2,...,λd).

$\sum$ v = μ v

Σ=Covariance matrix

V =Eigenvector μ = Eigen value

- Sort the eigenvectors by decreasing Eigen values and choose k eigenvectors with the largest Eigen values to form a d×k dimensional matrix W(where every column represents an eigenvector).
- Use this d×k eigenvector matrix to transform the samples onto the new subspace. This can be summarized by the mathematical equation $y = W^T * X$ (where x is a d×1-dimensional vector representing one sample, and y is the transformed k×1-dimensional sample in the new subspace.)

7.2. SAMPLE SIZE REDUCTION:

The original dataset has more than 5000 samples for each building, which would require a lot of efforts for data collection before building a model for another building in real life scenario. Therefore, we randomly select a subset of the original sample space to explore the effects of data size to the accuracy of the model. Data sizes we explored here in various experiments consists of 100, 200, 500, 1000, 2000 and 5000 samples.

7.3. MODEL SELECTION

We implement five classification methods in machine learning, including k-Nearest Neighbour (KNN), decision tree, Gradient Boosting, Logistic Regression and Support Vector Machine (SVM). All the four methods are applied with 10-fold cross validation to avoid overfitting.

8.KNN

KNN seems to be a good candidate for classification of this sort. It is due to the fact that KNN tries to make the classification by calculating the distance between features, while the intensity of various RSSI signals depends on the physical distance between Wifi source and mobile phones. In this case, closeness in feature space is a good indication of closeness in physical space.

8.1. SUMMARIZING THE KNN APPROACH

1. A positive integer k is specified, along with a new sample.

2. We select the k entries in our database which are closest to the new sample.

3. We find the most common classification of these entries.

4. This is the classification we give to the new sample.

8.2. FEATURES OF KNN

1. KNN stores the entire training dataset which it uses as its representation.

2. KNN does not learn any model.

3. KNN makes predictions just-in-time by calculating the similarity between an input sample and each training instance.

9. LOGISTIC REGRESSION

Logistic regression is a type of probabilistic statistical classification model. Logistic regression is a classification machine learning problem where the output of the machine is a discrete value. If the output is more than two discrete values it is multivariate logistic regression. Logistic Regression uses Gradient Descent algorithm for convergence.

Regression fits a good line but sometimes a curve too for the given labelled examples. The general equation of the line is **y=a*x +c** where m=slope and c=constant, and in this equation there is only one value of x (feature), but if there are more features we define the term **a*x+c** as **a*x 1+b*x2+c** or ($\boldsymbol{theta^T}$ *** x)** where x is a vector of all features and theta is a vector containing constants and initially the values of theta are zeros. Unlike linear regression where the output values are continous and h(x) the output of the machine it is a whole number. Logistic regression also uses the same cost function, but h(x) should not be continous rather it must have a value between -1 and 1. To make h(x) value between – 1 and 1 apply sigmoid function to ($\boldsymbol{theta^T}$ *** x**) .

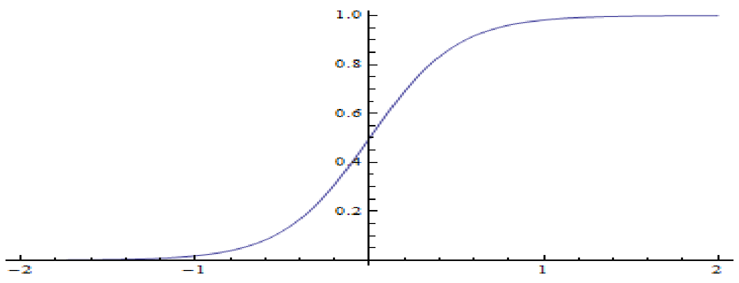
9.1. SIGMOID FUNCTION

A sigmoid function is a bounded differentiable real function that is defined for all real input values and has a positive derivative at each point. The logistic function has this further, important property, that its derivative can be expressed by the function itself (i.e if s(z) is sigmoid function then its derivative **s'(z) = s(z) * [s(z) – 1]**). Sigmoid function is defined as

s(z) =1/(1 + e^{-z} **).** Here **z =** $\boldsymbol{theta^T}$ *** X** where X is feature set

$$h(x) = 1/(1 + e^{-(theta^T * x)})$$

Sigmoid Function



9.2. GRADIENT DESCENT:

To make the value of h(x) nearly equal to y, we have to find the optimal value for theta. For finding optimal theta the cost function **J(theta) =** $\sum[ylog(h(x))$ – $(1-y)\log(1-h(x))]^2$ should continuously differentiated until convergence. α is learning rate, Lower the value of α more is the guarantee that the global minima value for theta is obtained generally the value of α is in the range **[0.001 , 0.1]** . Once theta is obtained the value ($theta^T$ * x) is calculated , if the value is greater than zero then the given example is positive example or else example is negative.

($theta^T$ * x) >= 0 => Positive Example

($theta^T$ * x) <0 => Negative Example

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Repeat {

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

} (simultaneously update all θ_j)

10. Support Vector Machine (SVM)

In machine learning, **support vector machines** 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 which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

When data are not labelled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups. The clustering algorithm which provides an improvement to the support vector machines is called **support vector clustering** and is often used in industrial applications either when data are not labelled or when only some data are labelled as a pre-processing for a classification pass.

It is basically used for classification purpose. It draws a decision boundary between positive and negative examples. Unlike logistic regression, here an example is considered as positive if **Z>1** and negative if **Z<-1.** So there are gutters between +1 and -1. SVM tries to draw a decision boundary that maximizes the width of the road that separates positive and negative examples. The main idea is to find widest separating street.

We can write the constraint:

$$\boldsymbol{y_i(w.x_i + b)} >= 1$$

y_i =+1 for positive examples and y_i =-1 for negative examples.

We classify unknown 'u' as follows:

$$\mathbf{f(u)} = \boldsymbol{w.u + b}$$

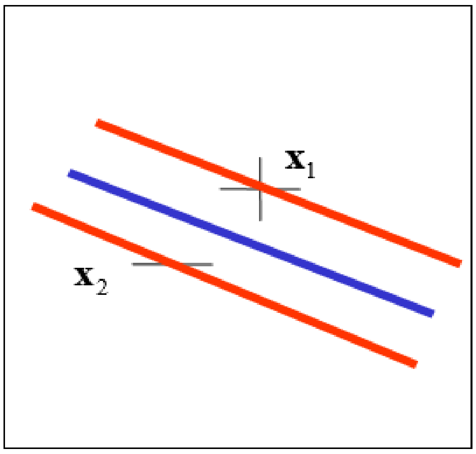
For all positive examples:

$$\mathbf{f(x_+)} = \boldsymbol{y_i(w.x_+ + b)} >= 1$$

For all negative examples:

$$\mathbf{f(x_-)} = \boldsymbol{y_i(w.x_- + b)} < -1$$

Distance between gutters is shown in figure.1:



$\boldsymbol{w}.\boldsymbol{x_1} + \boldsymbol{b}$ **=1** ------equation (1)

$\boldsymbol{w}.\boldsymbol{x_2} + \boldsymbol{b}$ **=-1** --------equation (2)

Subtracting equation (1) from equation (2)

$\boldsymbol{w}(\boldsymbol{x_1} - \boldsymbol{x_2})$**=2**

Dividing the above equation with unit vector

$\frac{\boldsymbol{w}}{||\boldsymbol{w}||}.\ (\boldsymbol{x_1} - \boldsymbol{x_2}) = \frac{2}{||\boldsymbol{w}||}$

Now to maximize the distance the factor ||w|| should be minimised.

For mathematical convenience it can be transformed as

$$\frac{1}{2}||\boldsymbol{w}||^2$$

Minimising the above equation can give maximum distance between the gutters.

One approach to find the minimum value is by using Lagrange's method. The resulting Lagrange multiplier that to optimise is:

$$L = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_i \boldsymbol{\alpha_i}(\boldsymbol{y_i}\,(\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b}) - \mathbf{1})$$

By partial derivatives we can get the values of alpha, w, b etc.

1. Partial derivative of L with respect to b :

   $$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{b}} = \mathbf{0}$$

   We obtain $\sum_i \boldsymbol{\alpha_i y_i} = \mathbf{0}$

2. Partial derivative of L with respect to w:

   $$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{w}} = \mathbf{0}$$

   We obtain $\boldsymbol{w} = \sum_i \boldsymbol{\alpha_i y_i x_i}$

The decision boundary is drawn by the equation:

h(x)= $\sum_i \boldsymbol{\alpha_i y_i K(x_i, x)} + \boldsymbol{b} \geq \mathbf{0}$

To classify an unknown *x* we should compute a kernel function $K(x_i, x)$ against each vector x_i

For positive gutter : **h(x)=** $\sum_i \boldsymbol{\alpha_i y_i K(x_i, x)} + \boldsymbol{b} = \mathbf{1}$

For negative gutter:

h(x)= $\sum_i \boldsymbol{\alpha_i y_i K(x_i, x)} + \boldsymbol{b} = \mathbf{-1}$

There are different types of kernels. Some of them are as follows

1.Linear Kernel

2.Polynomial Kernel

3.Radial Basis Function(RBF) or Gaussian Kernel.

In this project linear and Gaussian kernels are used.

Linear Kernel:

$$\boldsymbol{K}(\vec{\boldsymbol{u}},\vec{\boldsymbol{v}}) = \vec{\boldsymbol{u}}.\vec{\boldsymbol{v}}$$

If the examples are not separable they are transformed into another space.

$$\boldsymbol{K}(\vec{\boldsymbol{u}},\vec{\boldsymbol{v}}) = \emptyset(\vec{\boldsymbol{u}}).\emptyset(\vec{\boldsymbol{v}})$$

$\emptyset(\vec{u})$ that transforms input vectors into another dimension where they can be linearly separable.

Gaussian Kernel:

$$\boldsymbol{K}(\vec{\boldsymbol{u}},\vec{\boldsymbol{v}}) = \boldsymbol{exp}\left(-\frac{\|\vec{\boldsymbol{u}}-\vec{\boldsymbol{v}}\|^2}{2\sigma^2}\right)$$

Here σ^2 is the standard deviation of input vectors. It defines the steepness of the rise around the landmark. In 2D decision boundaries will be circles around positive and negative examples. As the value of σ^2 increases the feature falls to zero rapidly.

11. Decision Tree

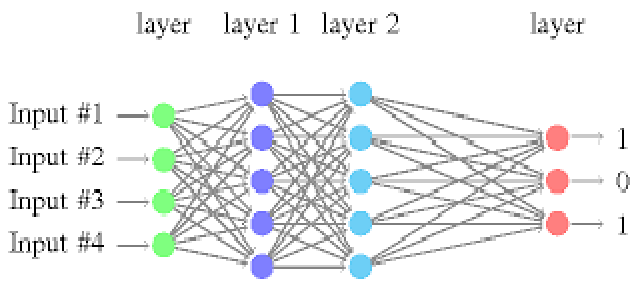
Decision tree is then implemented, which has the advantages of fast training process, easy interpretation and resistance to many irrelevant variables. But decision tree has the disadvantage of inaccuracy compared with KNN, even cross validation is used. In decision tree, two criteria are applied to prune the tree. One is cross validation and the other is one stand error. Surrogate splits are used in construction of the optimal tree as a missing value strategy, which encourages variables within highly correlated sets.

12. Boosting

Boosting is an ensemble technique in which the predictors are not made independently, but sequentially. To maintain most advantages of trees while dramatically improve accuracy, bagging algorithm could be a good choice. Here gradient boosting is chosen to improve the accuracy. Also, to handle missing data, we use surrogates to distribute instances. Best numbers of iterations (number of trees) are identified using cross validation and the depth for each simple tree is set to be four. This parameter could be further studied get a better accuracy.

13. Neural Networks :

Neural networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers' where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer' where the answer is output as shown below.



13.1. Layers:

Dropout:

Dropout is a regularization technique, which aims to reduce the complexity of the model with the goal to prevent over fitting. Using “dropout", you randomly deactivate certain units (neurons) in a layer with a certain probability p from a Bernoulli distribution (typically 50%, but this yet another hyper parameter to be tuned). So, if you set half of the activations of a layer to zero, the neural network won’t be able to rely on particular activations in a given feed-forward pass during training. As a consequence, the neural network will learn different, redundant representations; the network can’t rely on the particular neurons and the combination (or interaction) of these to be present. Another nice side effect is that training will be faster.

Dense Layers:

A dense layer represents a matrix vector multiplication. The values in the matrix are the trainable parameters which get updated during back propagation.

So you get a m dimensional vector as output. A dense layer thus is used to change the dimensions of your vector. Mathematically speaking, it applies a rotation, scaling, translation transform to your vector.

13.2. TRAINING MODEL:

The model that is used in out project consists of Input layer (Dense Layer), four hidden layers (Dropout layer , Dense layer , Dropout layer and Dense layer) and output Dense layer.

As there are 64 features (from PCA algorithm) the input Dense layer consists of 64 nodes and output layer consists of 5 nodes because of 5 class labels.

13.3. ACTIVATIONS:

Sigmoid Function:

$$\boldsymbol{sigmoid(x) = 1/(1 + e^{-1})}$$

ReLU Function:

$$\boldsymbol{ReLU(x) = 0\ if\ x \leq 0\ \ and\ \ x\ if\ x} > 0$$

13.4. COMPILATION:

Before training a model, you need to configure the learning process, which is done via the compile method. It receives three arguments:

• An optimizer. This could be the string identifier of an existing optimizer (such as rmsprop or adagrad), or an instance of the Optimizer class. See: optimizers.

• A loss function. This is the objective that the model will try to minimize. It can be the string identifier of an existing loss function (such as categorical_crossentropy or mse), or it can be an objective function. See: losses.

• A list of metrics. For any classification problem you will want to set this to metrics=['accuracy']. A metric could be the string identifier of an existing metric or a custom metric function.

14.MODULE :

The “scikit-learn” and “Keras” package in Python is specifically developed to handle many issues of machine learning and Neural Networks and also contains various in-built generalized functions that are applicable to all modelling techniques. Let us look at some of the most useful “sklearn” and “Keras” package functions.

14.1. Data Splitting

After loading the data, one of the first tasks that needs to be performed, is to split it into training and testing samples. Using "**train_test_split**" function. Syntax and other parameters supported by this function can be accessed by running the below function in Jupyter Notebook console.

And let's say development sample would 33% of observations of data and remaining observations into the validation sample.

```
from sklearn.cross_validation import train_test_split

X_train,X_test,Y_train,Y_test = train_test_split(X,Y,random_state=33)
```

14.2. FEATURE EXTRACTION:

PCA algorithm which takes all the features and transforms them into lower dimensional space in our case it is 64 dimension space. The sample code for the following is given below.

PCA

```
from sklearn.decomposition import PCA

logistic = linear_model.LogisticRegression()

pca = decomposition.PCA()

pipe = Pipeline(steps=[('pca', pca), ('logistic', logistic)])

pca=PCA(n_components=estimator.best_estimator_.named_steps['pca'].n_components)

pca.fit(X)

X1=pca.fit_transform(X)
```

14.3. MODEL BUILDING:

The below five lines builds model , fits data ,predicts labels for test data and finds accuracy. These five lines are common for most of the Machine Learning models.

KNN :

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=k)

knn.fit(X_train , Y_train)

Y_knn = knn.predict(X_test)

accuracy_score(Y_test , Y_knn)
```

LOGISTIC REGRESSION :

```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression(solver='lbfgs' , multi_class='multinomial')

lr.fit(X_train , Y_train)

Y_lr = lr.predict(X_test)

accuracy_score(Y_test , Y_lr)
```

SVM

```
from sklearn.svm import SVC

svc= SVC(C=1.0, kernel='rbf')

svc.fit(X_train ,Y_train)

Y_SVC =svc.predict(X_test)

accuracy_score(Y_test , Y_SVC)
```

DECISION TREE

```
from sklearn.tree import DecisionTreeClassifier

tr = DecisionTreeClassifier(criterion='gini')

tr.fit(X_train , Y_train)

Y_tr = tr.predict(X_test)

accuracy_score(Y_test , Y_tr)
```

RANDOM FOREST

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=10,criterion='gini')

rf.fit(X_train , Y_train)

Y_rf = rf.predict(X_test)

accuracy_score(Y_test , Y_rf)
```

EXTRA TREES CLASSIFIER

```
from sklearn.ensemble import ExtraTreesClassifier

ef = ExtraTreesClassifier(n_estimators=10,criterion='gini')

ef.fit(X_train , Y_train)

Y_ef = ef.predict(X_test)

accuracy_score(Y_test , Y_ef)
```

GRADIENT DESCENT

```
from sklearn.ensemble import GradientBoostingClassifier

gb = GradientBoostingClassifier(loss='deviance')

gb.fit(X_train , Y_train)

Y_gb = gb.predict(X_test)

accuracy_score(Y_test , Y_gb)
```

NEURAL NETWORKS

```
from keras.layers import Dense, Activation ,Dropout

from keras.models import Sequential

model = Sequential()

model.add(Dense(64, input_dim = X_train.shape[1] ,activation='relu'))

model.add(Dropout(0.2))

model.add(Dense(50 , activation = 'relu'))

model.add(Dropout(0.2))

model.add(Dense(25 , activation = 'relu'))

model.add(Dense(5 , activation = 'softmax'))
```

Compilation

```
model.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['a
ccuracy'])

model.fit(X_train,Y_train,batch_size=32,epochs = 10)

score = model.evaluate(X_test,Y_test,batch_size=32)
```

Different optimizers include SGD, RMSprop , Adagrad ,Adam.

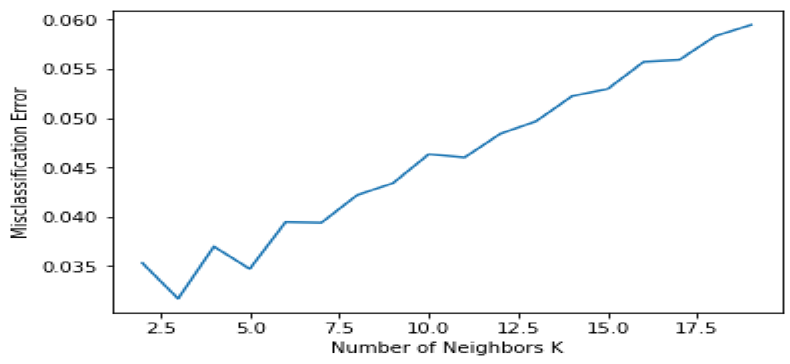
15. RESULTS :

We tested our models using data from three buildings separately. As mentioned before, at each building, we performed a PCA on the feature space to reduce its dimension, and randomly selected samples to perform a ten-fold cross validation on the four classification models. Both of the number of reduce dimension and the number of samples in the sample space are tunable for each model to achieve the least error. Each set of parameters was performed for 20 rounds and the error is averaged among those rounds to reduce noise.

KNN :

We first explore the number of nearest neighbours we need to consider for classifying a testing set. As shown in figure below, the error of classification increases with increasing of k. Then we look into the influence from the number of neighbours and the Misclassification Error we can observe that at KNN (k=4) an Elbow shape is formed where the algorithm performs well. The error reduces dramatically with increase of sample size, but not much improvement is seen from adding more principal features.

```
The optimal number of neighbors is %d 4
```



SVM :

SVM does not perform well for this problem. Here we explore both linear kernel and third order polynomial kernel, and decided to use polynomial kernel for better accuracy. This is because SVM tends to perform better for small dataset given enough margin data. Therefore, experiment in large feature and sample space will increase the difficulty of convergence for the optimization problem.

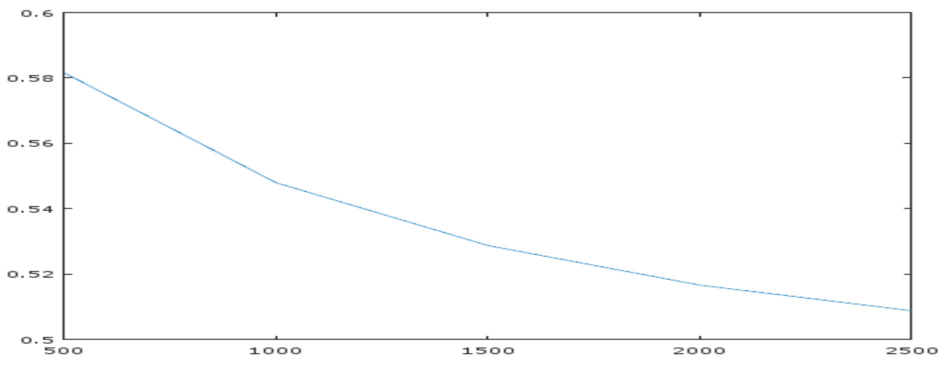
Boosting:

Three types of Boosting models are fitted based on the training data. They are Random Forest, Extra Trees Classifier with number of estimators as 10 and Gradient Boosting model with deviance loss, As all the models are based on more number of trees the prediction accuracy is best compared to SVM and Logistic Regression.

Logistic Regression:

In logistic Regression, the parameter alpha is known as learning rate. Learning rate is the main thing to get a good efficient algorithm, generally, the value of alpha ranges from **0.001 to 10**. Besides learning rate, the number of iterations in gradient descent also affects the algorithm, so how to choose a good value of alpha and iterations?

The possible way for this is with increase in the value of alpha and learning rate the cost obtained by alpha and iterations should decrease this ensures that the algorithm is performing better but how much? Gradient descent always chooses a value of alpha and iterations for which the cost is minimum. Hence while iterating for all the values of alpha and iterations there will be one combination for which the cost is minimum. Choose those values of alpha and iterations .Below is the graph between cost and iterations.



16. ACCURACY & METRICS

Efficiency in general terms is defined as the number of correctly predicted examples from the total examples. There can be four possibilities for any machine learning problem Correctly Identified **(True positive),** Correctly Rejected **(True Negative),** Incorrectly Identified **(False positive),** Incorrectly Rejected **(False Negative**).

Accuracy = (True positive + True negative)÷ (True positive + True negative + False positive + False negative)

Precision and recall

Two new metrics precision and recall both give a value between 0 and 1

Precision

How often does our algorithm false alarm?

Precision = true positives / (true positive + false positive)

High precision is good

Recall

How sensitive is our algorithm?

Recall = true positive / (true positive + false negative)

High recall is good

By computing precision and recall get a better sense of how an algorithm work

17. COMPARISIONS:

MODEL	ACCURACY
KNN	**96.62**
LOGISTIC	**87.64**
SVM	**64.87**
DECISION TREE	**89.82**
RANDOM FOREST	**94.72**
EXTRA TREE CLASSIFIER	**95.92**
GRADIENT BOOSTING	**92.71**

Neural Networks Optimizer Comparisons for 30 epochs

Optimizer	Accuracy
RMSprop	**92.82**
SGD	**93.84**
Adagrad	**94.30**
Adam	**94.58**

18. Conclusion

As demonstrated in this, the simplest KNN model gives good accuracy, given a relative small feature space and reasonable large data space. However, SVM performs poorly on this classification algorithm. Although one decision tree does not give satisfying result, bagging of multiple trees through gradient boosting could highly increase the prediction accuracy. To acquire high accuracy, while maintaining the capacity for predicting both small and large data set, we suggest the combination of KNN and gradient boosting for the indoor positioning system.

19. Future Work

Since gradient boosting is robust of missing value, the effect of missing value for current kNN model is to be investigated. Also, beyond the current models, tracking of moving user, type of phones and minimum number of Wifi sources required for accurate positioning will be explored in the future work.

20. References:

[1] Zhou, Junyi Shi, Jing. RFID localization algorithms and applications: a review. Journal of Intelligent Manufacturing, 20:, 695–707, 2009.

[2] Marques, Nelson Meneses, Filipe Moreira, Adriano. Combining similarity functions and majority rules for multi-building, multi-floor, WiFi positioning.

[3] Ching, William Teh, Rue Jing Li, Binghao Rizos, Chris. Uniwide WiFi based positioning system. Technology and Society (ISTAS), 2010 IEEE International Symposium on, 180–189, 2010.

[4] Chen, Mike Y Sohn, Timothy Chmelev, Dmitri Haehnel, Dirk Hightower, Jeffrey Hughes, Jeff LaMarca, Anthony Potter, Fred Smith, Ian Varshavsky, Alex/ Practical metropolitan-scale positioning for gsm phones. UbiComp 2006: Ubiquitous Computing, 225–242, 2006.

[5] http://scikit-learn.org/

[6] https://keras.io/

[7]https://archive.ics.uci.edu/ml/datasets/ujiindoorloc