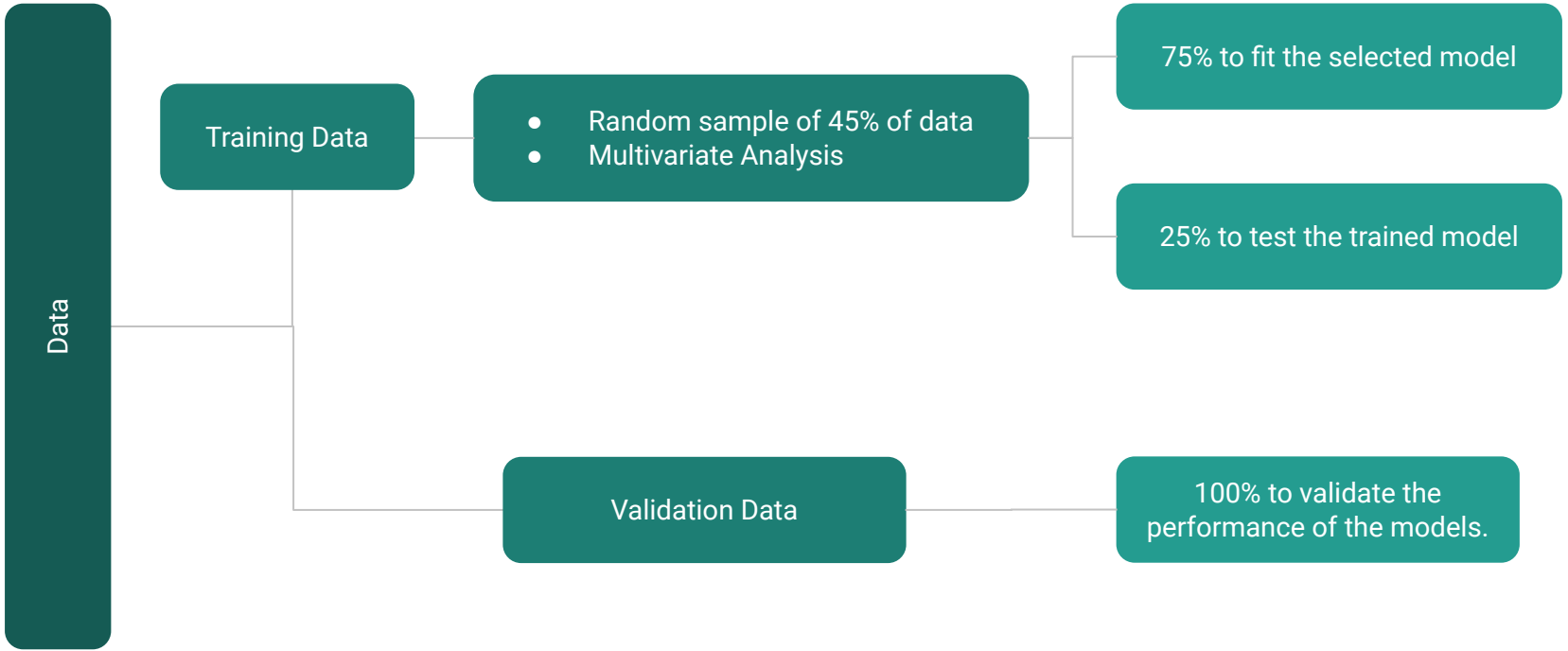
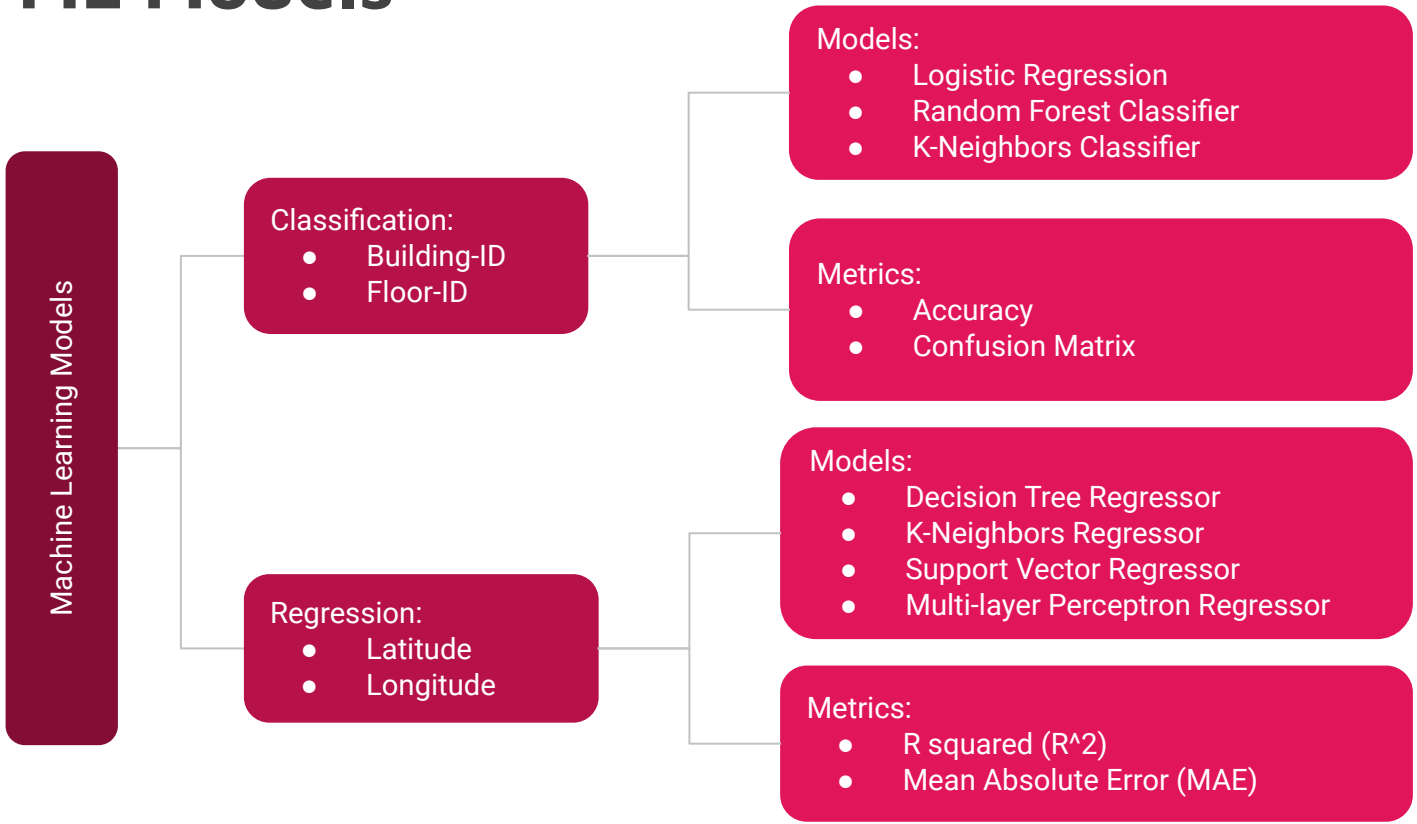
Evaluate ML Models for Wifi Locationing

by Ale f Benages.
Data analyst at **IOT Analytics**.

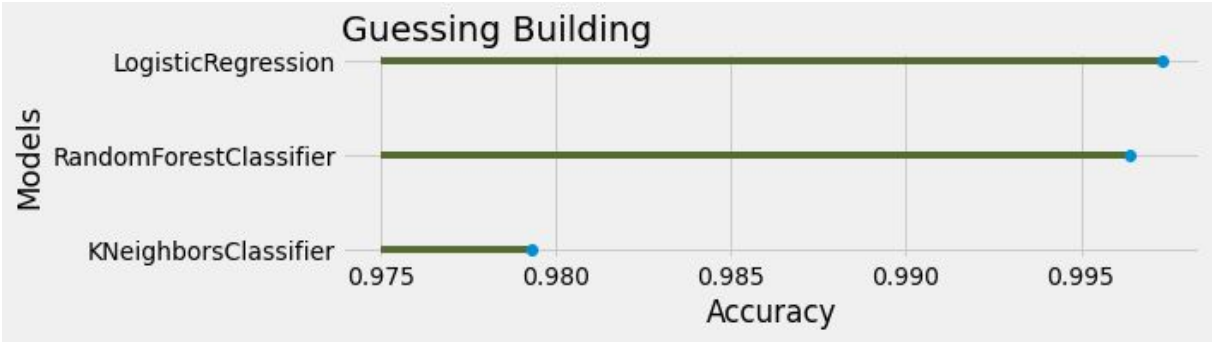
Pre-Processing Data



ML Models

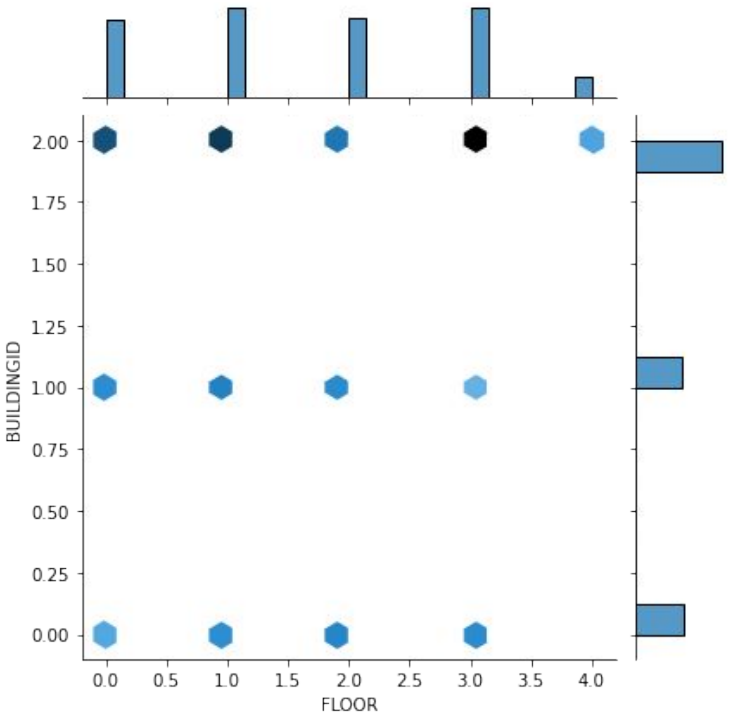

Machine Learning Models
Classification:
Building-ID
Floor-ID
Regression:
Latitude
Longitude
Models:
Logistic Regression
Random Forest Classifier
K-Neighbors Classifier
Metrics:
Accuracy
Confusion Matrix
Models:
Decision Tree Regressor
K-Neighbors Regressor
Support Vector Regressor
Multi-layer Perceptron Regressor
Metrics:
R squared (R^2)
Mean Absolute Error (MAE)

Guessing Building-ID



Building-ID it's a **categorical variable**, with 3 possibilities (Building 0,1,2), and **classification algorithms** were fitted.
The tree selected models performed great with the unseen data from the validation dataset. **KNeighborsClassifier** got an accuracy of **97.93%**, **Random Forest Classifier: 99,64%** and **Logistic Regression: 99,73%**.

Guessing Building-ID

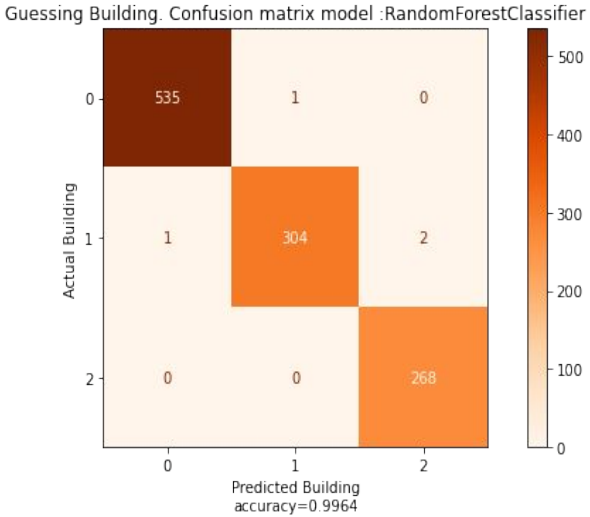


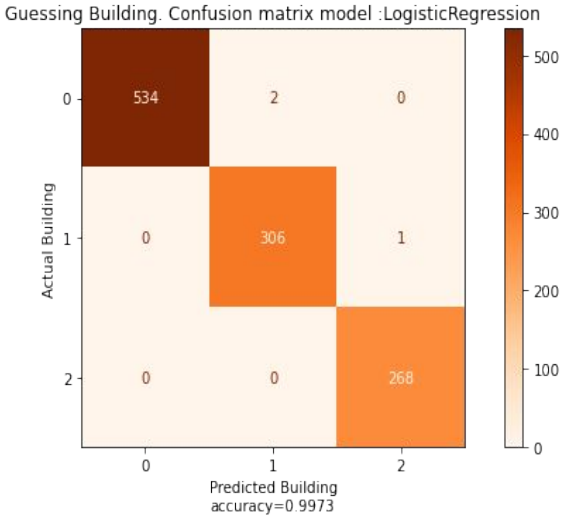
But perhaps accuracy isn't the best error metric choice. Classification accuracy fails on classification problems with a skewed class distribution.
This is the graph of the data used to train the models. It shows how much data contains each combination of Building/Floor.

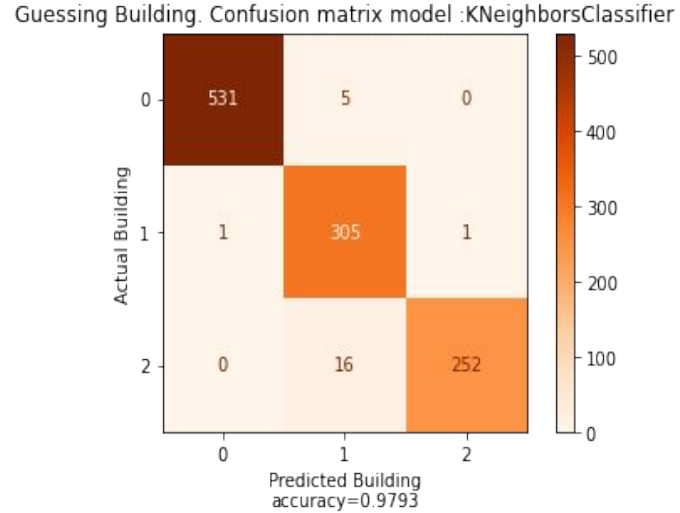
- BuildingID = 2 is the only one with a 4th floor. So it's logic that it contains very little data.
- BuildingID = 2 contains most of the examples.

So, another metric will produce meaningful information.

Guessing Building-ID



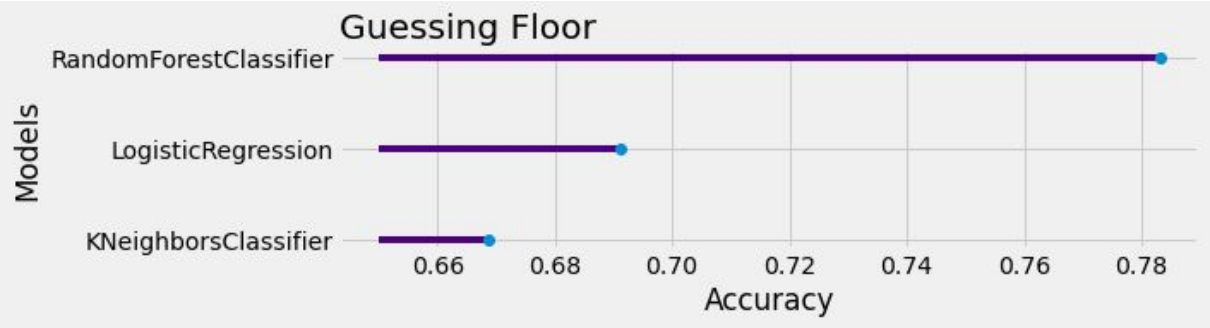




A better performance indicator could be a **Confusion Matrix**, were each row of the matrix represents the instances in an actual class while each column represents the instances in a predicted class. The diagonal contains the actual values that were correctly predicted by the algorithm, all the other values are misclassified observations.

The two first models (RF and LR) perform very well, with only 3 or 4 misclassified buildings. But again, **KNeighbors** is doing worse than the others.
I suggest choosing **Random Forest classifier**, as this model does not suffer from the overfitting (it takes the average of all the decision trees involved in the forest, cancelling the bias).

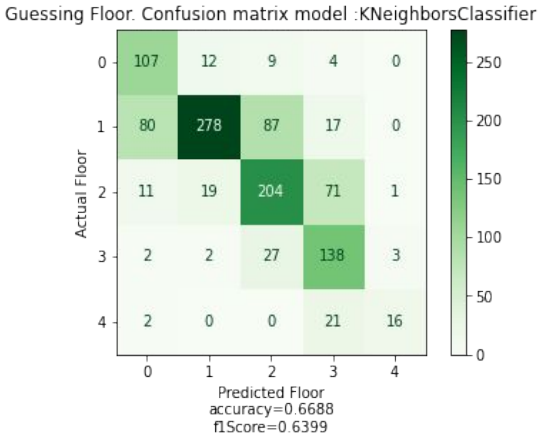
Guessing Floor-ID

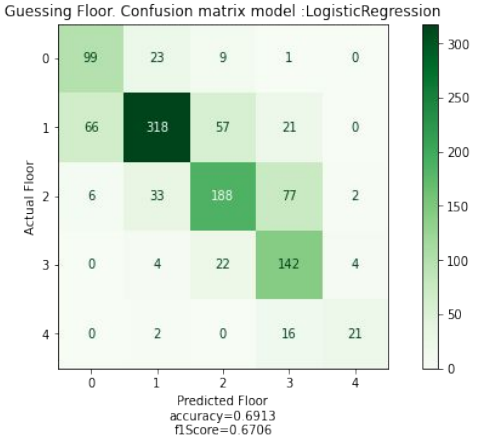


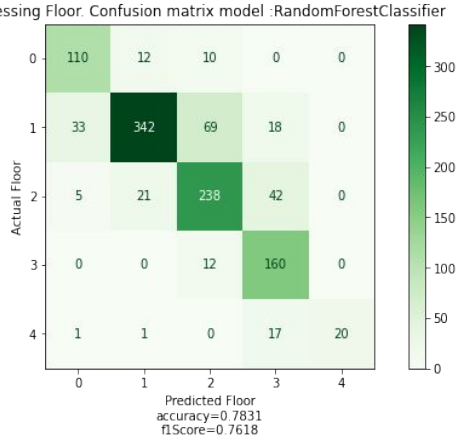
The same 3 models were also trained to guess the floorID. It's a **categorical variables**, and it's a **classification problem**. 5 possibilities were available (Floor 0,1,2,3,4).

In this case, the accuracy was not so impressive as in the previous guessing. Again **Confusion matrices** will be used to get more information.

Guessing Floor-ID





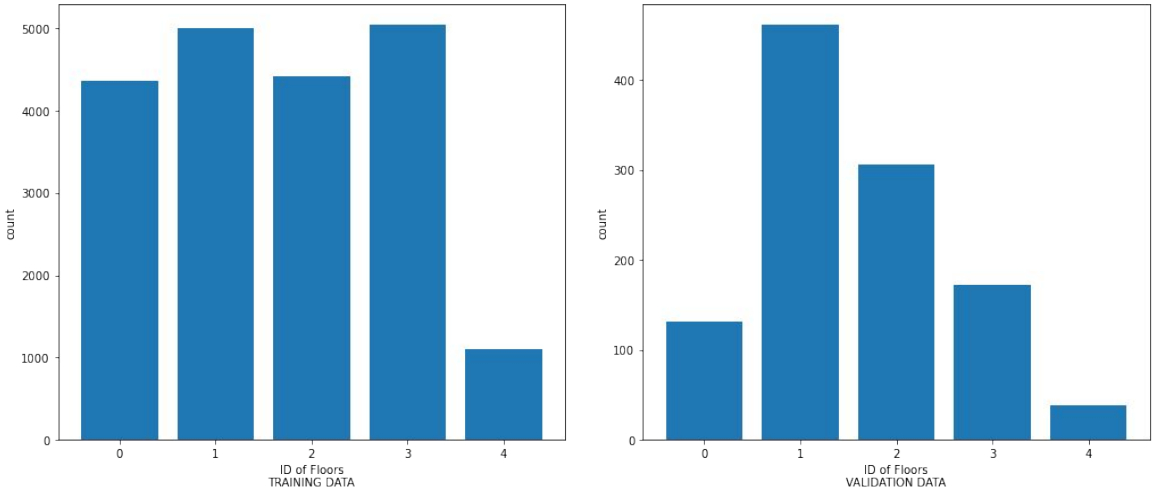


A confusion matrix is a better tool to judge an algorithm, than accuracy. But with multilabel problems (as in this case where there are 5 labels), it's not always easy to define which one is performing better by just watching the matrices. As we need to choose a model, we need one single measure to summarize model performance.

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. In the multi-class and multi-label case, this is the average of the F1 score of each class.

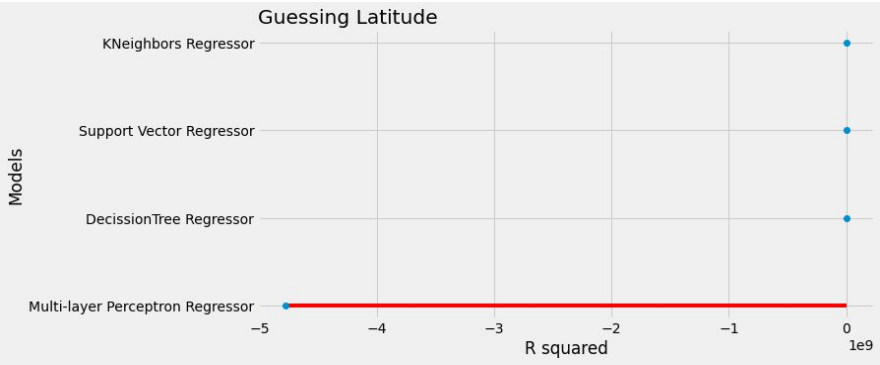
So, again **Random Forest Classifier** is performing better than others. with an **F1= 0.7618** (Also it's the model with the highest accuracy).

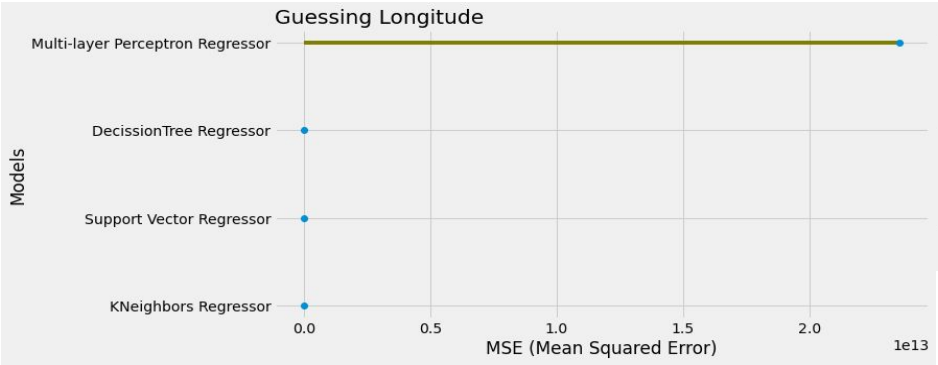
Guessing Floor-ID



I suspect that a possible cause of so many misclassified values (the ones outside the diagonal) is the great difference in the distributions of the samples between the train/test and the validation datasets.

Guessing Latitude`



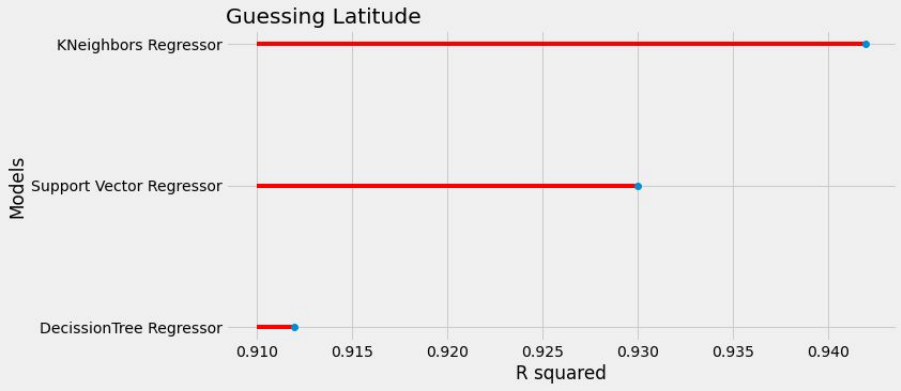


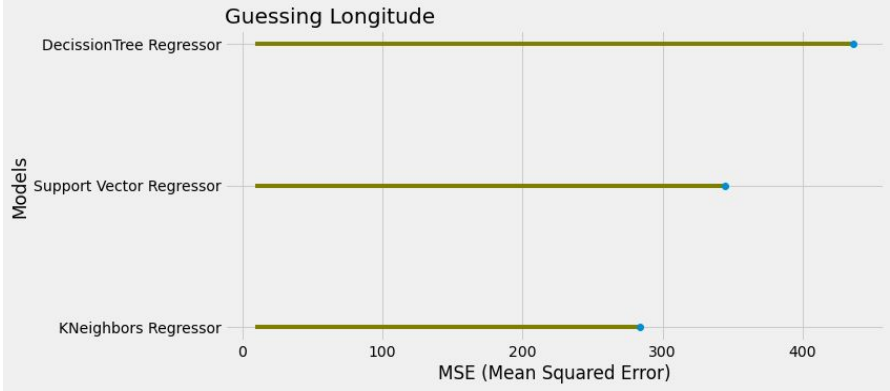
To guess Latitude (a continuous variable), four regression models have been trained: “**K-Neighbours Regressor**”, “**Support Vector Regressor**”, “**Decision Tree Regressor**” and “**Multilayer Perceptron Regressor**”.

Two error metrics were selected, R^2 and MSE. The **R^2** metric provides an idea of the goodness of fit of a set of predictions to the actual values. 1 means perfectly fit, and 0 no-fit at all. In this case all the models are fitting very well. In the other hand, **MSE** measures the average of the squares of the errors (the average squared difference between the estimated values and the actual value). It’s always a positive value.

I wasn’t able to tune the parameters and hyper parameters and get decent result with **MLP**. With the unseen data it performed even worst.

Guessing Latitude

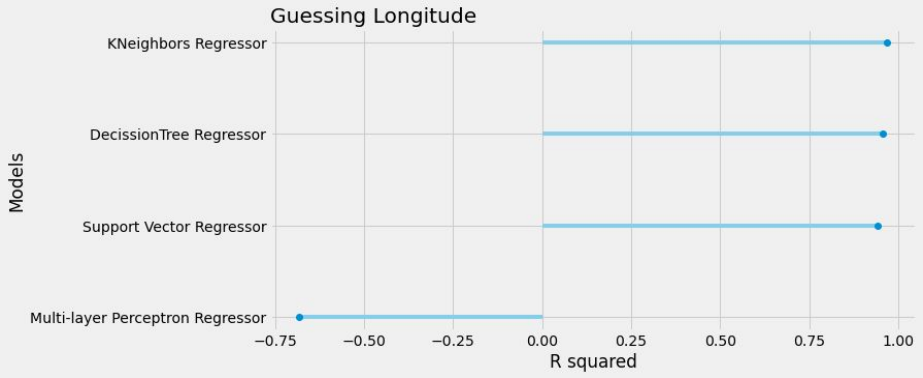


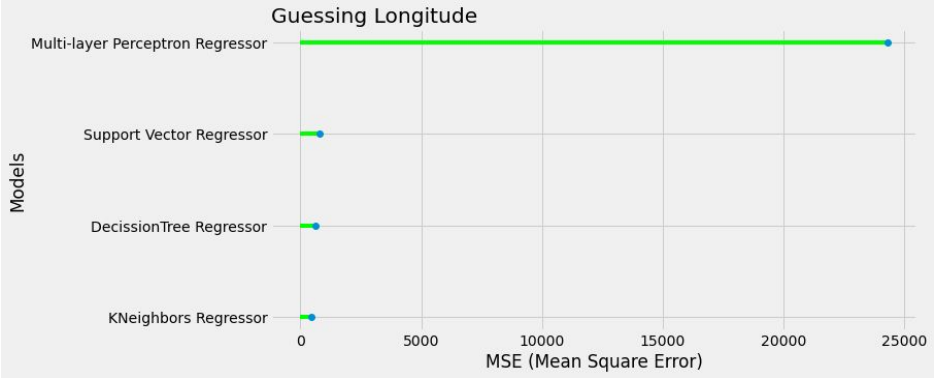


Without MLP things are clearer: the three remaining models are performing quite well with the unseen data.

The ideal situation is a model with an r^2 closer to 1, and a MSE closer to 0. In this case **KNeighborsRegressor** wins in both metrics.

Guessing Longitude

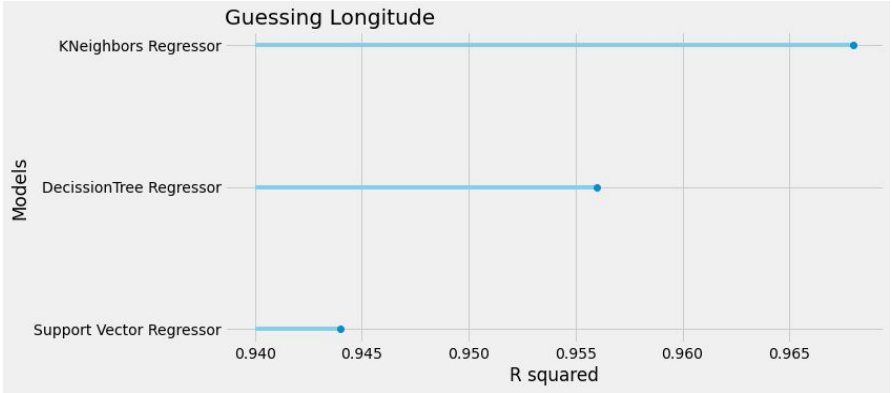


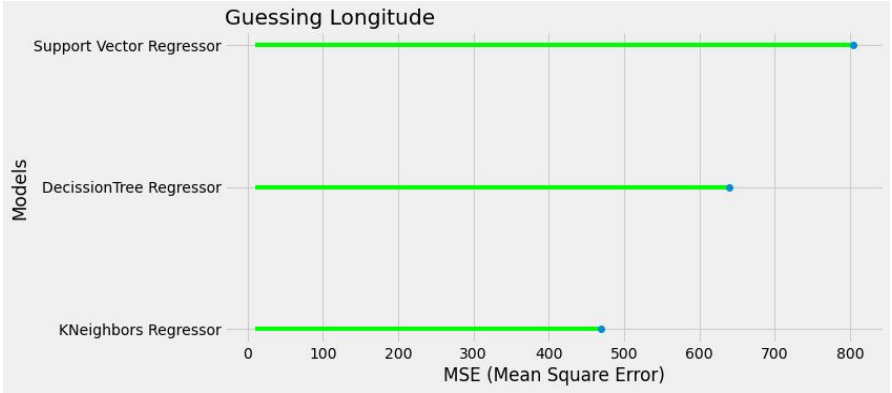


To guess the Longitude, the same 4 modes have been trained, and the same metrics have been used.

Again MLP is performing unacceptable, and will be removed to simplify the analysis.

Guessing Longitude





Recalling that the ideal situation of a r^2 closer to 1, and a MSE closer to 0.

Once again **KNeighborsRegressor**,wins in both with an **R^2= 0.968** and **MSE=468.91**

Summarizing

01	BUILDING ID	• **Model: Random Forest Classifier** • CV = 10 folds • Tuned Decision Parameters (by Grid Search CV): {'max_depth': 60, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 20}
02	FLOOR ID	• **Model: Random Forest Classifier** • CV = 10 folds • Tuned Decision Parameters (by Grid Search CV): {'max_depth': 60, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 40}
03	LATITUDE	• **Model: KNeighborsRegressor** • CV = 10 folds • Tuned Decision Parameters (by Grid Search CV): {'leaf_size': 1, 'n_jobs': -1, 'n_neighbors': 2, 'weights': 'distance'}
04	LONGITUDE	• **Model: KNeighborsRegressor** • CV = 10 folds • Tuned Decision Parameters (by Grid Search CV): {'leaf_size': 1, 'n_jobs': -1, 'n_neighbors': 2, 'weights': 'distance'}

Possible Improvements

- Choose another combination of models/parameter/hyper parameters to guess Floor-ID.
- Perhaps combine the FloorID with another feature, like SpaceID, to increase the accuracy of this feature.
- Plot some latitude-longitude examples to see how they fall over the map. I imagine they could:
    - Understand better how the model is working, and possibly reduce it's error.
    - Plotting the misclassified ones, could provide some useful information about the areas of a weak strength.