Facebook check-in prediction

Rida Zainab
rzainab@stevens.edu

Ao Xu
axu@stevens.edu

***Abstract*—Research has shown that brand recommendations from someone with whom one has a personal relationship weigh more than most other sources of advertising. Social media serves as a primary medium today for sharing such information first-hand by users themselves. Every second, 3.3 million new posts appear on Facebook and almost half a million on Twitter and a significant part of these social media postings are check-ins into different places. Understanding the patterns and details of such huge amount of data requires special set of tools and skills such as machine learning and statistics. Therefore, public places such as restaurant, shopping malls and customer centered business ventures are becoming increasingly interested in investing into leveraging machine learning and artificial intelligence methods to understand and optimize the huge amount of data generated by their users. In 2016, Facebook introduced a competition on the popular machine learning data platform Kaggle. Based on a given dataset and a standard mode of evaluation, the goal of the competition was to predict which place a person would like to check in to.**

I. Introduction

User check-ins have gained popularity on social media platforms off late. Facebook and Kaggle launched a machine learning engineering competition in 2016 based on prediction of user check-in. The goal of this competition is to predict which place a person is likely to check in to. For the purposes of this competition Facebook created an artificial world consisting of more than 100,000 places located in a 10km by 10km square. For a given set of coordinates, the task is to return the place ID of the most likely place. Data was fabricated to resemble location signals coming from mobile devices, giving a flavor of what it takes to work with real data complicated by inaccurate and noisy values as inconsistent and erroneous location data can disrupt experience for services like Facebook Check In.

II. Data Overview

The dataset consists of approximately 29 million observations and roughly 380,000 unique places where the location (x, y), accuracy (this feature was referred to as ”quality” in the presentation to avoid confusion, and timestamp is given along with the target variable, the check in location represented as place IDs of the 380,000 unique places. an important thing to consider is that here is no concept of a person or a user in this dataset; all the observations are events, not people. The details of the 4 features are not revealed and it is part f the competition to guess what they could corespond to. It was quickly established by the kaggle machine learning community that time is measured in minutes and could thus be converted to relative hours and days of the week, etc. Accuracy was by far the hardest input to interpret, it was considered as the location accuracy for an observation or the nature of the GPS or WIFI used to connect to internet to make the check-in. The features x and y ranged from 0 to 10 so it was naturally assumed that these 2 features correspond to the location. The place ID is the place of business, this is also the target we are required to predict. As the dataset proved to be too heavy for repeated experimentaton and computation, we decided to take a smaller sample out of the whole data. In order to do that, we selected data from places which and x and y location in the 1x1 km area out of the total 10x10 km data. This also helped in getting an automatically scaled x and y feature from 0 to 1 km. In order to get better and faster performance of our models, we split the sampled data further into two halves for separate computation on each half.

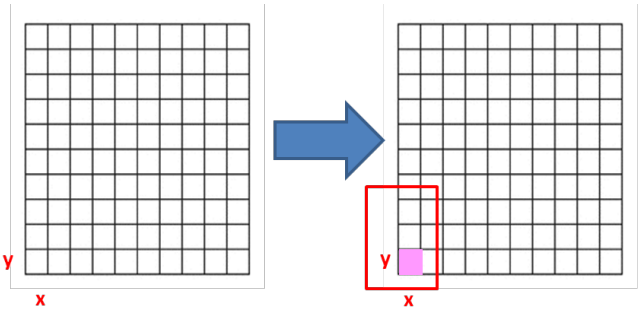

Fig. 1. Data Sampling

III. Methodology

A. Pre-processing

In order to better understand the dataset, we developed lots of plot to show how these features related to each other. Since our whole project is a multi-labels classification problems, checking the labels is an important step for the feature engineering part. In our sample dataset, we calculated top 10 place ids which appears more frequently than others. And we plot 10 subplots to see how these 10 places appears in the 10x10 map. Firstly, through a series of plots we observed that the behaviour of features in the full 10x10 km data and our sampled 1x1 km data was comparable, which means that the models we develop can be attuned and applied to other grids

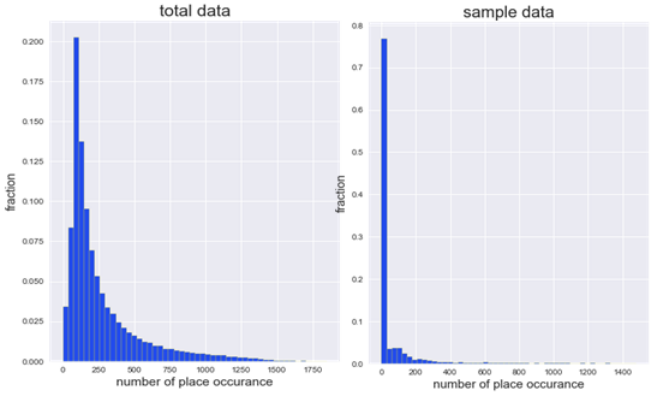

Fig. 2. Overview of data

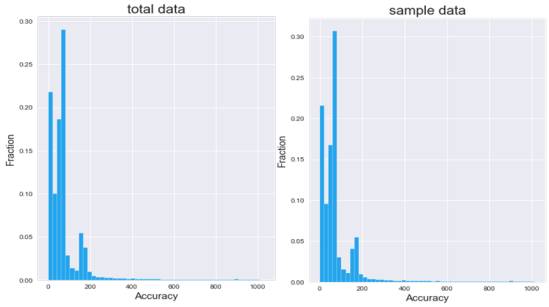

Fig. 3. Feature Distribution: Accuracy

Fig. 4. Feature Distribution: Time

in the area and they should give the same range of accuracy which we observe for 1x1 km data. We plotted the clases alongside their respective sample size and we observed two ineresting aspects of the data (1) some place IDs occur only a few times with some place IDs having just one sample. We oberved that 3912 place ID (3912 different classes) out of 5917 classes in the 1x1 km data occur 5 or less than 5 times. This means that the chance that all 5 or less than 5 samples of many of these 3912 classes will all fall into the test set or the training set with no sample in the other corresponding test or train set is quite high which resulted in high inaccuracy in some o ur earlier comptations, (2) there is a wide difference between the sample size of the classes. Some classes have thousands of samples and some samples have only one single sample which means that we have unbalanced classes, (3) We observed useful pattern in the timestamp and x and y location features but we did not find much variation between the accuracy feature.

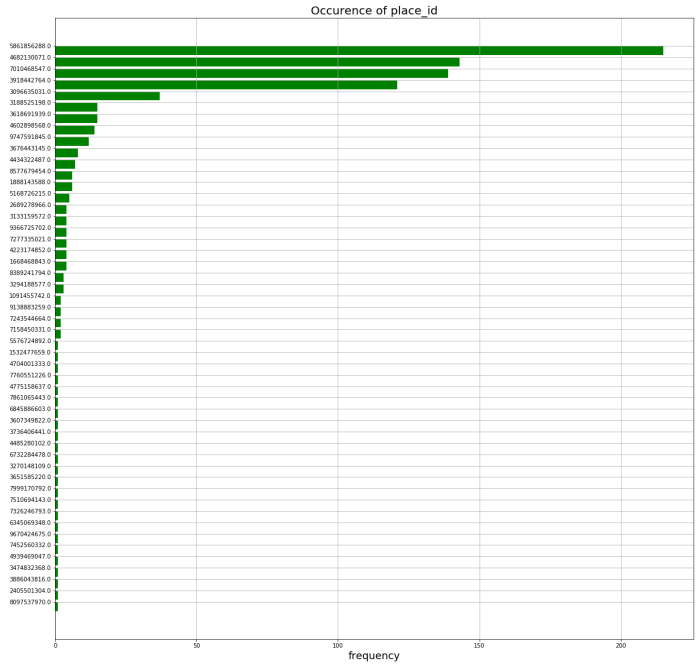

Fig. 5. Class frequency

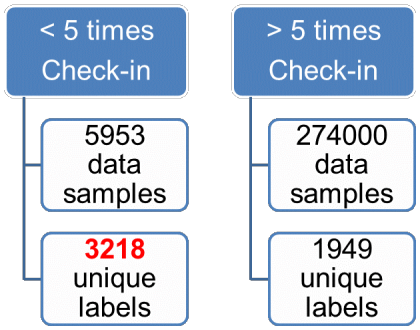

Fig. 6. Unbalanced classes

The top 10 places appear as different clusters in the map. We can consider that different labels can be treated as different clusters based on x and y location. The distribution for these most popular 10 places shows that the variation along x is much more than that along y for most of the places, the larger variance along x could be due to the fact that the check-ins are made on the vertically placed lanes of

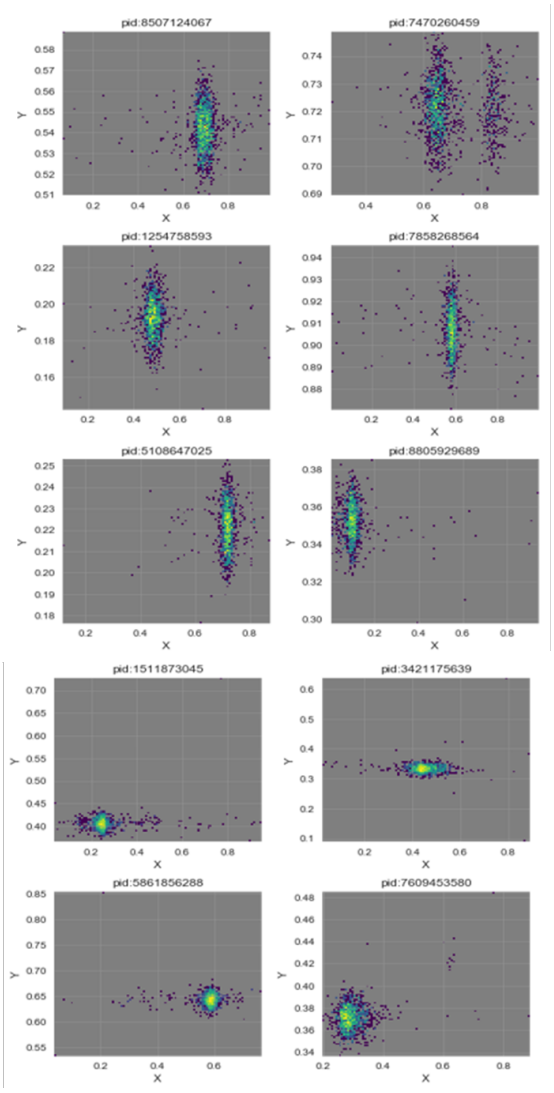

Fig. 7. Feature Distribution: Top 10 places x and y plot

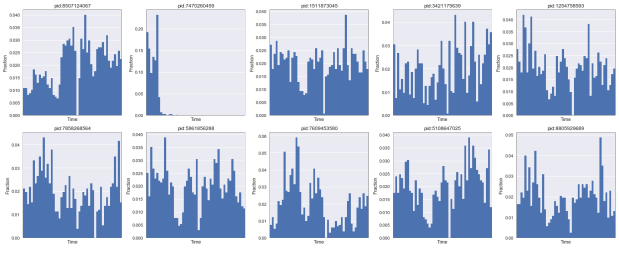

Fig. 8. Feature Distribution: top 10 places time

the grid.

The definition of time and its scale in the dataset is intentionally left vague as per the specifications, we used kernel density estimation function to get a better idea of how the popularity of a place varies with time. We picked the top 10 places and see that these places can vary significantly with time and we further explore this in the feature design.

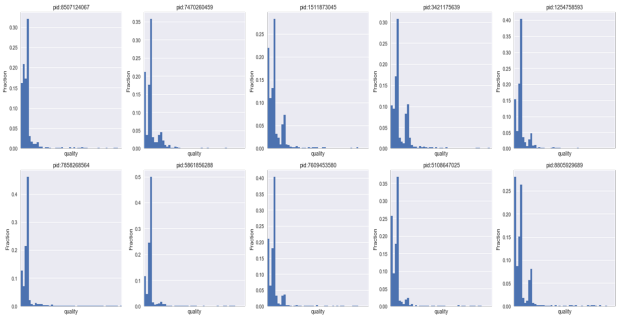

Fig. 9. Feature Distribution: Top 10 places accuracy(quality) feature

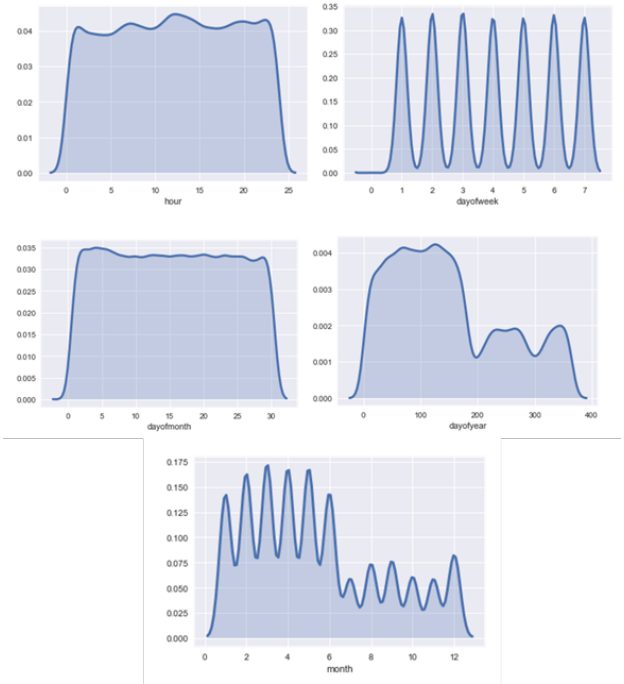

Fig. 10. Time Distribution

The peaks in the figure give valuable information that certain places have higher/lower check-in rate at specific time periods. From the time values in the dataset it can be inferred that the time spans across 24 hours of the day, 7 days of a week, 12 months and 365 days of a year. The reason certain places might be popular at certain time values could be due to the nature of the place example, parks have higher number of visitors during the day while clubs have during the night. And based on this new finds, we also explore such periods characteristics in the feature design.

To deal with the problem of large number of labels and unbaanced classes we oversampled the data. The details are in the following section.

1) Oversampling and under sampling: To deal with the problem of unbalanced classes, oversampling and

undersampling techniques were used to increase the occurence of low frequency class labels. Oversampling and undersampling in data analysis are techniques used to adjust the class distribution of a data set (i.e. the ratio between the different classes/categories represented). Oversampling and undersampling are opposite and roughly equivalent techniques. They both involve using a bias to select more samples from one class than from another. Ror our dataset, we found oversampling to be more useful. We picked samples out of low frequency classes and added them again to the data to increase the sample size of previously low frequency classes. However, we still had classes with below 100 samples which we dropped to reduce the sample ratio between difference classes. [10]

Another technique commonly used while oversampling is to add very small magnitude random gaussian noise in the randomly picked data samples from the low frequncy classes and add these noisy samples to the data to model real-world data. This is similar to creating artificial samples.

B. Feature Engineering

Based on these plots and properties of these features, we decided to create more new additional features. Firstly, for the time features, we divided time into hour,day of the week, day of month, the month and day of the year. It was obesrved that time feature did not vary much during the day.

For location x and y features, we used K-Nearest Neighbours to calculate 10 nearest neighbours and used features based on distance from those neighbours but that did not give encouraging results so we dropped that technique and calculated mean of x and y feature and the ratio of x and y.

For Accuracy feature, we tried to split accuracy in a range based on the plots and make additional feature based on the range of accuracy but that did not yield fruitful result so we dropped it and used the accuracy feature as it is.

Next we computed k-means based on all features. the detail of k-means is in the section below.

1) K-means: In the feature engineering part, we initially use k-means algorithm to createdifferent number of clusters from 5-100 based on the location(x,y), and treat it as new feature to help us label the observations better. The k means clustering is one of the simplest and popular unsupervised machine learning algorithms. It starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative calculations to optimize the positions of the centroids. However, we did not observe any improvemnt in the classification results so we dropped the k-means feature.[3]

C. Classification

Based on the nature of the data we used KNN, Naïve Bayes, Decision Tree, AdaBoost, Random Forest and Stacking for the purpose of classification. The above algorithms were considered a suitable fit for this problme primalrily because of the high number of classes which ruled out linear supervised classification algorithms such as SVM, LDA etc.

The performance of each of these classification algorithms is discussed below.

1) K-Nearest Neighbours: The K-Nearest Neighbour (KNN) algorithm is among the simplest of all machine learning algorithms. It is a non-parametric method used for classification and regression. The input consists of the k closest training examples in the feature space. For the purpose of classification, the output is a class membership. An object is classified by a majority 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.[6]

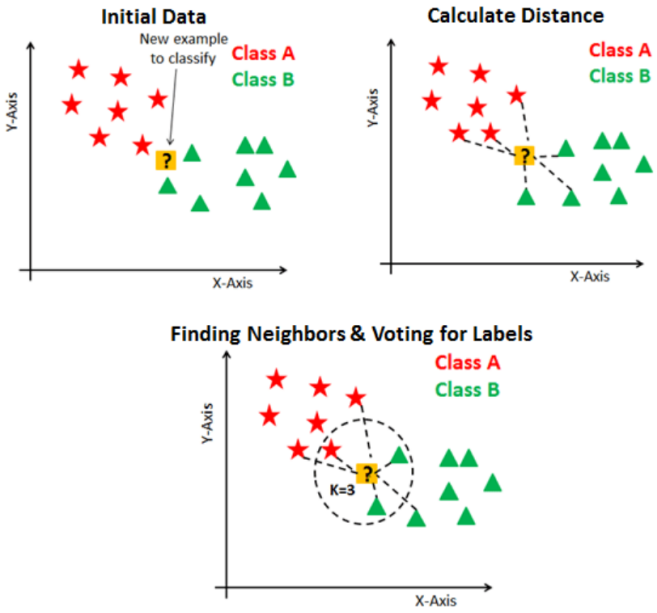

Fig. 11. KNN

For our dataset, we tried a range of KNN neighbours from 5-500. Changing the number of k neighbours had little effect on the final classificatio acuuracy after 20 k nearest neighbours. A peculiarity of the k-NN algorithm is that it is sensitive to the local structure of the data.

2) Naïve Bayes: Naive Bayes classifiers are a family of probabilistic classifiers based on Bayes' theorem with strong (naive) independence assumptions between the features. Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form

expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.Bayes theorem provides a way of calculating the posterior probability, P(c—x), from P(c), P(x), and P(x—c). Naive Bayes classifier assume that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence. [8]

$$P(c|x) = \frac{P(x|c) * P(c)}{P(x)}$$

where $P(c|x)$ is the posterior probability of class (target) given predictor (attribute)

$P(c)$ is the prior probability of class

$P(x|c)$ is the likelihood which is the probability of predictor given class

$P(x)$ is the prior probability of predictor

3) Decision Tree: Decision Tree algorithm belongs to the family of supervised learning algorithms. The decision tree algorithm can be used for solving both regression and classification problems.The general motive of using Decision Tree is to create a training model which can be used to predict class or value of target variables by learning decision rules inferred from training data. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label. The algorithm starts by placing the best attribute of the dataset at the root of the tree. It then split the training set into subsets. Subsets should be made in such a way that each subset contains data with the same value for an attribute. The above steps are repeated on each subset until leaf nodes are found in all the branches of the tree. [2]

4) 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. This machine learning method is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features, and produces inspectable models. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model. [5]

For our dataset, we used random forest with 100 estimators (100 trees) and depth of 25. The features were weighted based on the sample size of their respective classes.

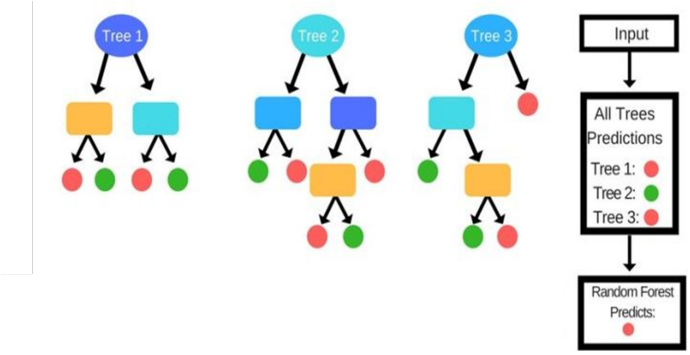

Fig. 12. Random Forest

5) AdaBoost: AdaBoost, short for Adaptive Boosting, is a popular machine learning meta-algorithm used in conjunction with many other types of learning algorithms to improve performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner. Every learning algorithm tends to suit some problem types better than others, and typically has many different parameters and configurations to adjust before it achieves optimal performance on a dataset. When used with decision tree learning, information gathered at each stage of the AdaBoost algorithm about the relative 'hardness' of each training sample is fed into the tree growing algorithm such that later trees tend to focus on harder-to-classify examples.[4]

For our dataset we used AdaBoost with both Decision Tree and Random Forest classifiers of varying depth of trees. It was observed that the AdaBoost performed better with Random Forest as compared to Decision Tree.

6) Neural Network: In machine learning, a neural network (NN) is an interconnected group of artificial neurons that uses a mathematical or computational model for information processing. Neural networks are non-linear statistical data modeling or decision making tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. An artificial neural network involves a network of simple processing elements (artificial neurons) which can exhibit complex global behavior, determined by the connections between the processing elements and element parameters. The utility of artificial neural network models

lies in the fact that they can be used to infer a function from observations and also to use it. However, they require a large amount of data for training, are slow to train and not as robust as other supervised learning classifiers such as Random Forest. [7]

For our dataset, we used NN with 10 hidden layers and 50 hidden units.

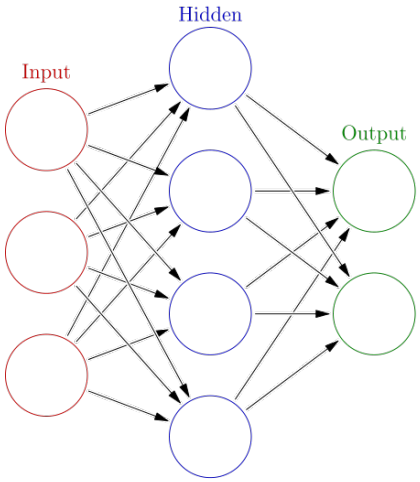

Fig. 13. Neural Network: Example

7) Stacking: Stacking is an ensemble learning technique that combines multiple classification or regression models via a meta-classifier or a meta-regressor. The base level models are trained based on a complete training set, then the meta-model is trained on the outputs of the base level model as features.The base level often consists of different learning algorithms and therefore stacking ensembles are often heterogeneous. [9]

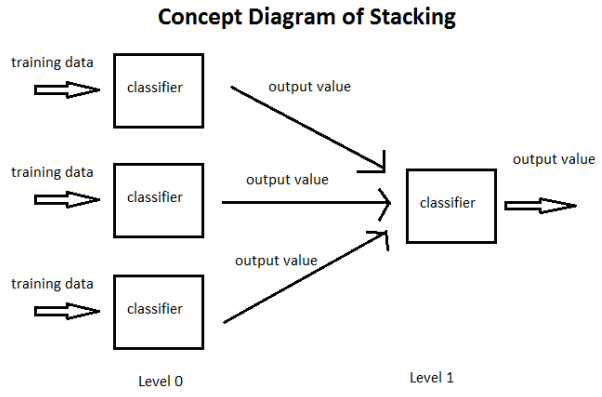

Fig. 14. Stacking

For our dataset, we used Naive Bayes, KNN, and Decision Tree for stacking and the final clasifier was Random Forest.

TABLE I
CLASSIFICATION ACCURACY

Algorithm	Accuracy
Naive Bayes	0.52
KNN	0.36
Decision Tree	0.73
Random Forest	0.82
AdaBoost	0.80
Stacking	0.67

TABLE II
EFFECT OF OVER SAMPLING

Algorithm	Accuracy before sampling	Accuracy after sampling
Decision Tree	0.61	0.73
Random Forest	0.64	0.82
AdaBoost	0.67	0.67

However, KNN reduced the accuracy of the model so we dropped it.

IV. RESULTS

The accuracy table shows that Random Forest (RF) gives the best results followed closely by AdaBoost. We used random RF with a variety of estimators and depth of tree and found out that 20 to 100 tress and depth of 20 to 25 gives best results. However, for AdaBoost due to memory limitation we used Random Forest with nly 20 trees and 20 estimators. Decision tree appears to be the third best algorithm. The table shows that ensemble methods tend to better fit the data and predict accurate labels as opposed to non-ensemble methods. Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. A machine learning ensemble consists of only a concrete finite set of alternative models, but typically allows for much more flexible structure to exist among those alternatives. Empirically, ensembles tend to yield better results when there is a significant diversity among the models. We incorporated diversity into the data by over-sampling low frequency class labels which reduced the error.

To further improve the performance of ensemble methods, different clustering techniques such as Locally Adaptive Clustering (LAC) can be used as features [1]. LAC is similar to k-means. K-means clustering uses all input features

with equal relevance which may not always be effective. LAC computes clusters in subspaces spanned by different combinations of dimensions via local weightings of features, which is more effective when features show different variance. Such as the x and y feature in our data - for some classes x shows more variation and for other classes y shows more variation. Compard to other features, accuracy does not show any difference amongst classes. Such behaviour is better captured by clustering algorithms such as LAC.

Another important observation is that this classification problem resembles text classification problem as both have a large number of output space. Therefore, algorithms such as Naive Bayes is expected to perform better. However, we did not observe a good accuracy for Naive Bayes. We suspect the relatively low accuracy of Naive Bayes is due to the fact that it considers the features independent of each other, which is not the case for our data. The additional features did not contribute to the performance of accuracy of Naive Bayes and it remained more or less the same.

V. Conclusion

This paper demonstrated the classification problem of an unbalanced dataset with labels in the order of thousands. This is an unconventional classification problem because of three main reasons (1) The large number of classes, (2) unbalanced classes, some classes have very few ocuurences and some have total samples in the order of hundreds and thousands, (3) less number of features considering the huge size of the dataset. We only a selected a small part of the whole dataset for computation. Were we to use the whole data, the accuracy is expected to reduce. Even with the full 10x10 km data, a significant number of place IDs have just sample which would result in many inaccurate predictions. Therefore, the oversampling and discarding very low frequncy samples might still prove to be useful.

Acknowledgment

The authors would like to thank the organizers of Facebook V Kaggle Competition for providing the dataset.

References

[1] C Domeniconi, “Locally Adaptive Metrics for Clustering”, IBM TJ. Watson Research Center
[2] J.R. QUINLAN, “ Induction of Decision Trees” , Centre for Advanced Computing Sciences, New South Wales Institute of Technology, Sydney 2007, Australia
[3] T Kanungo, “An Efficient k-Means Clustering Algorithm: Analysis and Implementation”
[4] R E. Schapire, “Explaining AdaBoost”
[5] G Biau,“Analysis of a Random Forests Model”
[6] Yun-lei Cai, Duo Ji ,Dong-feng Cai, “A KNN Research Paper Classification Method Based on Shared Nearest Neighbor”.
[7] Christian Szegedy, “Deep Neural Networks for Object Detection”
[8] Andrew Y. Ng, “On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes”.
[9] Funda Güneş, “Stacked Ensemble Models for Improved Prediction Accuracy”.
[10] James Bryant, “Oversampling and Undersampling”