NORTHEASTERN UNIVERSITY

CS6220 DATA MINING TECHNIQUES

Project Report

Food borne Illness Analysis

Srijit Ravishankar

Surupa Tushar Chatterjee

December 10, 2018

Contents

1 Introduction **2**

2 Exploratory Data Analysis **3**
2.1 Data Cleaning 3
2.2 Visual Analysis of Data 3
2.3 Correlation Analysis of Data 4

3 Models **6**
3.1 Task 1 : Predict if a case is a confirmed or suspected case of food borne illness 6
3.2 Baseline Models - Classification 7
3.2.1 K Nearest Neighbors 7
3.2.2 Logistic Regression 8
3.2.3 Ada Boost 9
3.2.4 Gradient Boosting 10
3.2.5 Random Forest 11
3.3 Task 2: Predict the number of people affected by Food Borne Illnesses 12
3.4 Baseline Models - Regression 12
3.4.1 Linear Regressor 12
3.4.2 Ridge Regressor 13
3.4.3 Decision Tree Regressor 13
3.4.4 Random Forest Regressor 14
3.4.5 Gradient Boosting Regressor 15

4 Model Building **15**
4.1 Task 1: Predict confirmed/suspected case of food borne illness - classification 15
4.2 Task 2: Predict number of food borne illness - Regression 16

5 Results **16**
5.1 Predict confirmed/suspected case of food borne illness 16
5.2 Task 2: Predict number of food borne illness 17

6 Model Evaluation **18**
6.1 Task 1: Predict confirmed/suspected case of food borne illness . 18
6.2 Task 2: Predict number of food borne illness 18

7 Conclusion **19**

8 Future Work **19**

1 Introduction

Over the past decade there have been numerous diseases that have been detected and reported to have caused outbreaks that led to many deaths. Many of these diseases have been identified to have origins from food that a person eats. The Center for Disease Control and Prevention (CDC) estimates roughly 1 in 6 Americans (48 million people) get sick, 128,000 are hospitalized, and 3,000 die of food borne diseases each year. A food borne disease outbreak occurs when two or more people get the same illness from the same contaminated food or drink. While most food borne illnesses are not part of a recognized outbreak, outbreaks provide important information on how germs spread, which foods cause illness, and how to prevent infection.

Here we use the data set that contains data on food borne disease outbreaks reported to CDC from 1998 through 2016. Data fields include year, state (outbreaks occurring in more than one state are listed as "multistate"), location where the food was prepared, reported food vehicle and contaminated ingredient, etiology (the pathogen, toxin, or chemical that caused the illnesses), status (whether the etiology was confirmed or suspected), total illnesses, hospitalizations, and fatalities. In many outbreak investigations, a specific food vehicle is not identified; for these outbreaks, the food vehicle variable values are blank.

We aim to apply data mining techniques that can help discover interesting patterns in this data set and apply algorithms to predict the patterns of food borne illness outbreaks, identify the key ingredients/sources in food that lead to infections, specific regions/areas across U.S where the risk of food borne diseases is high/low, where the patient consumed the food(Restaurants/Home etc.), specific period of the year when a particular disease is more prominent and spreads, kind of pathogens or germs that leads to a disease or specific toxin/chemical that contributes to the disease, method of the food preparation that may lead to the outbreak.

We plan to explore the above-mentioned data set and use additional data sets such as a data set that would contain features pertaining to what time of the year a particular food pathogen may be more active and more prone to grow and spread, food materials that contain specific toxic materials and quantities of those toxins that can lead to infections/diseases and so on.

The goal of our project is to apply supervised machine learning techniques to build predictive models to predict whether a case is suspected, confirmed or no food borne illness and the number of people that may be affected with food borne illnesses.We also intend to address the following questions - Whether a particular food can cause food borne disease?, Which pathogens, toxins or chemicals in the food lead to diseases?, Food Combinations and their potential infection source(pathogen/toxin/chemical), Deem a food to be fatal on the basis of number of hospitalizations and fatalities?

2 Exploratory Data Analysis

2.1 Data Cleaning

We noticed that there were a lot of data points in the data set with multiple values. For example, Location had values [Fair/Festival, Restaurant]. This data point was exploded to create two data points, one with Location as Fair/Festival and other with Restaurant. Such data points in the data sets were handled in a similar fashion.

We also aimed at understanding the structure of the outbreaks.csv data set, where we found that data contains mostly categorical data.The shape of the data set is elaborated as follows:

	Year	Month	State	Location	Food	Ingredient	Species	Serotype/Genotype	Status	Illnesses	Hospitalizations	Fatalities
0	1998	January	California	Restaurant	NaN	NaN	NaN	NaN	NaN	20	0.0	0.0
1	1998	January	California	NaN	Custard	NaN	NaN	NaN	NaN	112	0.0	0.0
2	1998	January	California	Restaurant	NaN	NaN	NaN	NaN	NaN	35	0.0	0.0
3	1998	January	California	Restaurant	Fish, Ahi	NaN	Scombroid toxin	NaN	Confirmed	4	0.0	0.0
4	1998	January	California	Private Home/Residence	Lasagna, Unspecified; Eggs, Other	NaN	Salmonella enterica	Enteritidis	Confirmed	26	3.0	0.0

Figure 1: Outbreaks

After this, the next big roadblock was the many missing data that had to be handled. A Mean/Mode methodology was employed to handle the missing data. The numerical features that had missing values were replaced with the mean of their respective columns and the categorical features that had missing values were replaced with the mode of their respective columns.

2.2 Visual Analysis of Data

Various graphs were plotted to understand the food borne illness data set. From the data and Figure 2, it was found that the Pathogen which is the major cause of food borne illnesses was $\boldsymbol{Salmonellaenterica}$. followed by $\boldsymbol{NovoVirusGenogroup1}$.

It was also observed from Figure 3 that food borne illnesses predominantly occur when there is an intake of $\boldsymbol{FinFish}$ and/or $\boldsymbol{Pork}$.

Over 11,094 cases of reported food borne illnesses were picked up after eating in a $\boldsymbol{Fair/Festival}$, followed by $\boldsymbol{Restaurant}$ as seen in Figure 4.

The risk of food borne illnesses seems to be high in the state of $\boldsymbol{California}$ as seen in Figure 5.

We also plotted a Time Series graph to understand the period of the year when food borne illnesses were predominantly reported. It was found that during the month of August, reported cases of food borne illnesses, both confirmed and suspected cases of Food borne illness were at peak.

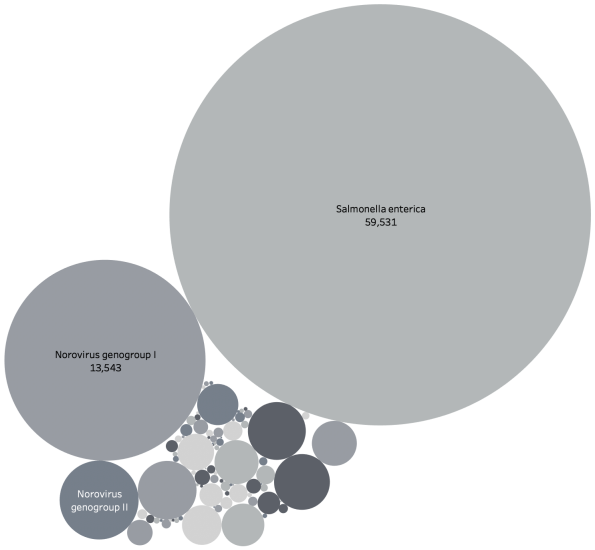

Figure 2: Species causing Food borne illnesses

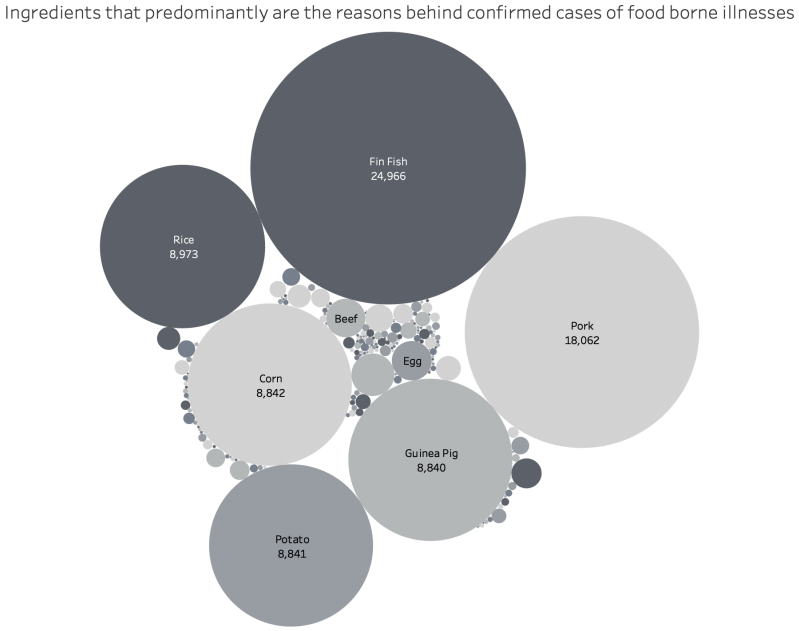

Figure 3: Major Ingredient causing Food borne illnesses

2.3 Correlation Analysis of Data

Correlation is defined as a statistical association, although in common usage it refers to how close two variables are to having a linear relationship with each other. Correlations are useful because they can indicate a predictive relationship that can be exploited in practice.

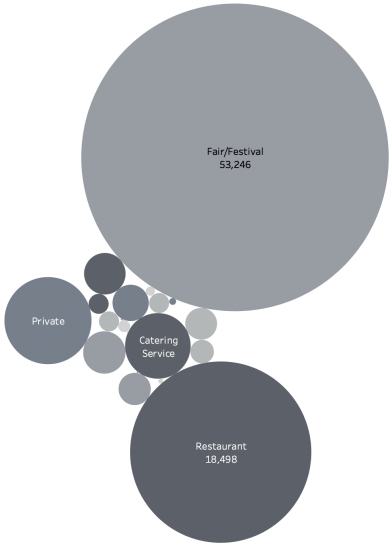

Figure 4: Major Location where a Food borne illnesses was picked up

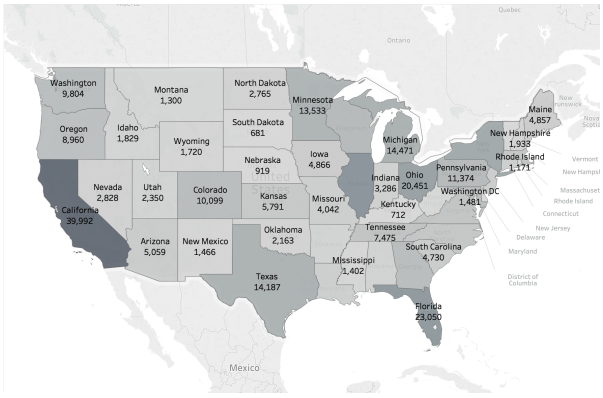

Figure 5: State statistics for confirmed cases of food borne illnesses reported

A correlation graph was plotted to analyze the relationship/association across all the features in the data set. This was done to uncover any relationship between any two features to prevent any multicollinearity affecting the models that would be built. Figure 7 shows the correlation plot.

We can infer from these statistics that more precautions and safety measures can be taken or employed in the state of California, primarily during the month of August, to ensure the well-being of its citizens. Also, proper immunizations can be taken against the Species of Salmonella Enterica and NovoVirusGenogroup1 to avoid any risk of contracting any food borne illnesses.

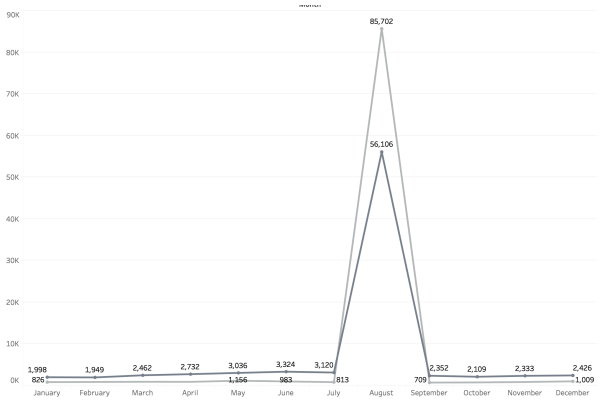

Figure 6: Period of the year when food borne illnesses were at peak

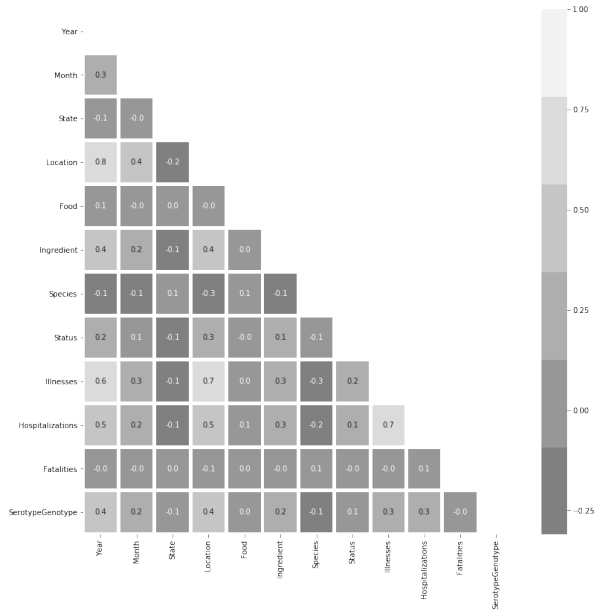

Figure 7: Correlation of various features in the data set

3 Models

3.1 Task 1 : Predict if a case is a confirmed or suspected case of food borne illness

To predict if a reported case of food borne illness is a suspected or confirmed case of illness, a distribution of the target variable was plotted to check for any imbalance as seen in Figure 8. Since it had little or no imbalance, we proceeded with building the baseline models.

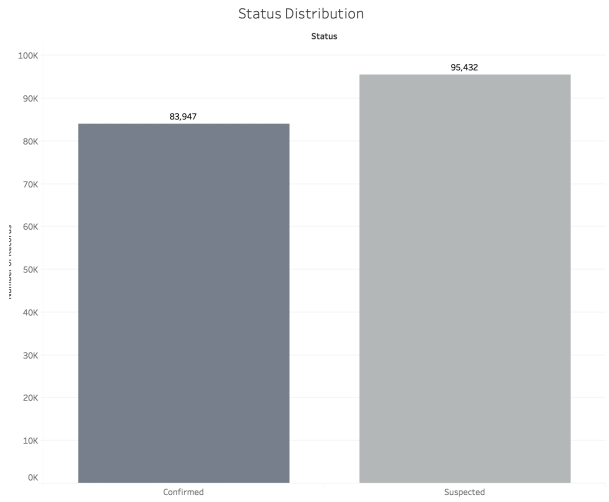

Figure 8: Distribution of the target variable : Status

3.2 Baseline Models - Classification

Since this is a classification problem, various models were implemented with default model parameters to choose a best model for the problem. We used the following models as baseline to select our best model:
1. K Nearest Neighbors
2. Logistic Regression
3. Ada Boost Classifier
4. Gradient Boosting Classifier
5. Random Forest classifier

3.2.1 K Nearest Neighbors

K Nearest neighbors classifier is probably the most simplest algorithm that can be used for classification task. We chose to start with KNN because it provided us with the basic intuitions required to understand how it would work for this classification problem.
KNN classifier classifies an unknown data sample by finding the most common class among the K closest data samples found through distance computation between new data point and each sample point.Each data sample in the K closest example casts a vote and class with the highest number of votes is predicted for the unknown sample.There are four basic steps to the algorithm as :

1. Calculates the distance between the new point and every other point.
2. Sorts the distances computed in ascending order of distance values and selects the top K samples from the sorted list.
3. Uses majority voting to predict the class of the new data point.

This baseline model gave us an accuracy of 59.75%

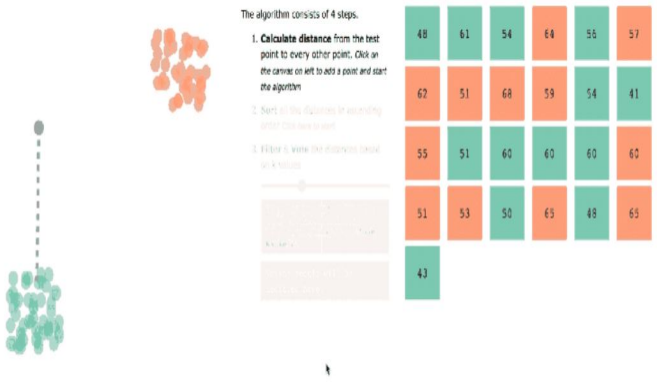

Figure 9: Step1 : KNN Distance computation

```
Model Report
Accuracy : 0.5975
AUC Score: 0.645335
              precision    recall  f1-score   support

           0       0.57      0.56      0.57     16796
           1       0.62      0.63      0.62     19080

   micro avg       0.60      0.60      0.60     35876
   macro avg       0.60      0.60      0.60     35876
weighted avg       0.60      0.60      0.60     35876
```

Figure 10: KNN Classification Report

Since the accuracy obtained was not satisfactory, we moved on to try other baseline models.

3.2.2 Logistic Regression

We chose logistic regression as our second choice since we are trying to do a binary classification and logistic regression is an widely known algorithm that performs quite well on a number of problems.

Logistic regression uses linear equation with independent predictors to predict a value.This predicted value can be anywhere in the range of negative infinity to positive infinity.This predicted value is then squashed between a range of [0,1] using sigmoid function.Thus there are values that are close to 0,that evaluate to class 0 for classification, whereas the values that are close to 1.
1. Linear Equation of x can be written as :

$$z = \theta_0 + \theta_1.x_1 + \theta_2.x_2 + ...$$

2. The output z is then passed to sigmoid function to get values in the range [0,1] given by :

$$h = g(z) = \frac{1}{1 + e^{-}z}$$

Sigmoid function can be visualized as seen in Figure 9.

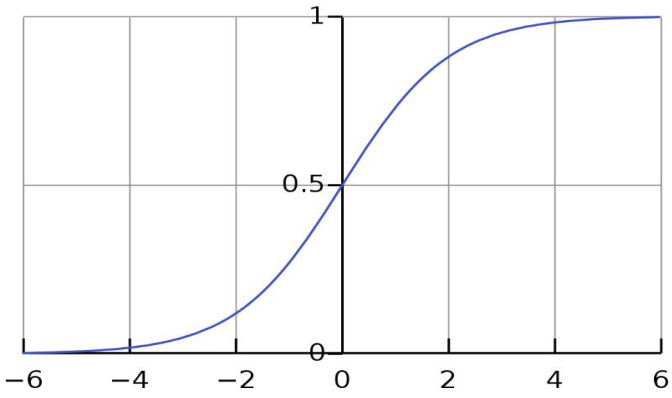

Figure 11: sigmoid Function

The Logistic Regressor with the default parameters, gave an Accuracy of 0.6396 and AUC Score of 0.624 as seen in Figure 12.

Model Report
Accuracy : 0.6396
AUC Score: 0.624351

	precision	recall	f1-score	support
0	0.76	0.34	0.47	16796
1	0.61	0.90	0.73	19080
micro avg	0.64	0.64	0.64	35876
macro avg	0.68	0.62	0.60	35876
weighted avg	0.68	0.64	0.61	35876

Figure 12: Performance of Logistic Regression

3.2.3 Ada Boost

The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data. The predictions from all of them are then combined through a weighted majority vote (or sum) to produce the final prediction.

The data modifications at each so-called boosting iteration consist of applying weights w_1, w_1, ..., w_n to each of the training samples. Initially, those

weights are all set to $w_i = 1/\mathrm{N}$, so that the first step simply trains a weak learner on the original data. For each successive iteration, the sample weights are individually modified and the learning algorithm is reapplied to the re-weighted data. At a given step, those training examples that were incorrectly predicted by the boosted model induced at the previous step have their weights increased, whereas the weights are decreased for those that were predicted correctly. As iterations proceed, examples that are difficult to predict receive ever-increasing influence. Each subsequent weak learner is thereby forced to concentrate on the examples that are missed by the previous ones in the sequence[5].

The number of weak learners is controlled by the parameter n_estimators. The learning_rate parameter controls the contribution of the weak learners in the final combination. By default, weak learners are decision stumps. Different weak learners can be specified through the base_estimator parameter. The main parameters to tune to obtain good results are n_estimators and the complexity of the base estimators (e.g., its depth max_depth or minimum required number of samples to consider a split min_samples_split)[5].

The performance of this model on our data set was relatively better than knn and logistic regression. The Ada Boost Classifier, with the default parameters, gave an Accuracy of 0.6638 and AUC Score of 0.6567 as seen in Figure 13.

```
Model Report
Accuracy : 0.6638
AUC Score: 0.656718
              precision    recall  f1-score   support

           0       0.84      0.35      0.49     16796
           1       0.62      0.94      0.75     19080

   micro avg       0.66      0.66      0.66     35876
   macro avg       0.73      0.64      0.62     35876
weighted avg       0.72      0.66      0.63     35876
```

Figure 13: Performance of Ada Boost Classifier

3.2.4 Gradient Boosting

Gradient Tree Boosting (GBT) is a generalization of boosting to arbitrary differentiable loss functions. GBT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems. Gradient Tree Boosting models are used in a variety of areas including Web search ranking and ecology. The various parameter available to tune are the same as for Ada boost algorithm.

The advantages of GBT are - Natural handling of data of mixed type Predictive power and the Robustness to outliers in output space (via robust loss functions) and the disadvantage of GBT is Scalability, due to the sequential nature of boosting it can hardly be parallelized. [10]

The performance of this model on our data set was relatively better than knn, logistic regression and the Ada Boost Classifier. The Gradient Boosting

Classifier, with the default parameters, gave an Accuracy of 0.67 and AUC Score of 0.6666 as seen in Figure 14.

```
Model Report
Accuracy : 0.67
AUC Score: 0.666600
              precision    recall  f1-score   support

           0       0.85      0.36      0.50     16796
           1       0.63      0.95      0.75     19080

   micro avg       0.67      0.67      0.67     35876
   macro avg       0.74      0.65      0.63     35876
weighted avg       0.73      0.67      0.64     35876
```

Figure 14: Performance of Gradient Boosting Classifier

3.2.5 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[6][7]. Random decision forests correct for decision trees' habit of over-fitting to their training set[8].

Also, Each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. In addition, when splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features. As a result of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree) but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model[9].

Random Forest Classifier gave the best performance on our Data set. With the default parameters, it gave an Accuracy of 0.6638 and AUC Score of 0.6567 as seen in Figure 15.

```
Model Report
Accuracy : 0.6852
AUC Score: 0.665214
              precision    recall  f1-score   support

           0       0.92      0.36      0.52     16796
           1       0.63      0.97      0.77     19080

   micro avg       0.69      0.69      0.69     35876
   macro avg       0.78      0.67      0.64     35876
weighted avg       0.77      0.69      0.65     35876
```

Figure 15: Performance of Random Forest Classifier

3.3 Task 2: Predict the number of people affected by Food Borne Illnesses

In order to predict the number of people that may be affected due to food borne illness we used various regression models to understand which among the various independent variables(features) are highly related to the dependent variable(illnesses) and how they impact the dependent variable for predictions. In the process to identify the best model that would fit our data well and make accurate predictions we tried a number of baseline models and then fine tuned the parameters of the best performing model to improve the accuracy.

3.4 Baseline Models - Regression

1. Linear Regressor
2. Ridge Regressor
3. Decision Tree Regressor
4. Gradient Boosting Regressor
5. Random Forest Regressor

3.4.1 Linear Regressor

We started with the most basic model that can be used to predict the value of the dependent variable using number of independent variables by finding the best fit line for the data and using that line to predict the value of the dependent variable for any new given values for the independent variables.

The best fit line is chosen such that prediction error(all data points) is as low as possible. Prediction error is the distance between the point and the regression line

$$Y(Pred) = \theta_0 + \theta_1.x_1 + \theta_2.x_2 + \theta_3.x_3 + ...$$

We choose the theta values such that they minimize the error. We use squared error as a metric to evaluate the model with a goal to obtain the best line that reduces the error.Error is calculate as below:

$$Error = \sum_{i=1}^{n} (y_i - y_{Pred})^2$$

The performance of this model on our outbreaks data set, gave an variance score of 66.17%. We also calculated the coefficient estimates to check which features are highly correlated and impact the value of the "Illness" variable both positively and negatively, and found that "Hospitalizations" and "Locations" are positively highly correlated and "Fatalities" is negatively correlated. The following figure gives an useful visualization:

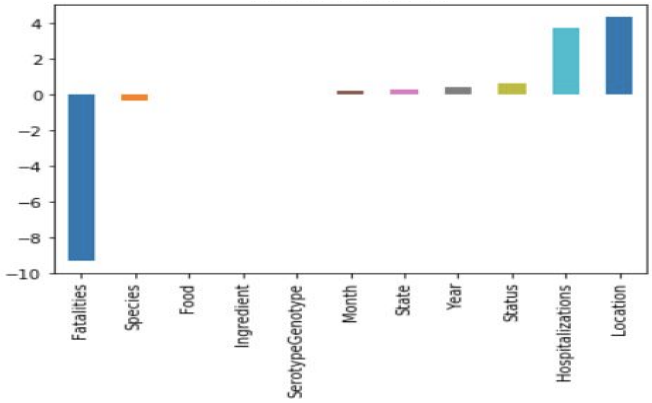

Figure 16: Model Coefficients

We also wanted to check other baseline models and find the best model among them to be able to achieve a better accuracy score. We therefore moved on to another linear model

3.4.2 Ridge Regressor

As the model coefficients "Location" and "Hospitalizations" have a very high value, we wanted to ensure that we do not have a case of multicollinearity where the predictor variable "Illnesses" can be linearly predicted using each of the independent variables with a substantial degree of accuracy. Multi-collinearity is the existence of near-linear relationships between the independent variables. Ridge regression uses parameter called as "alpha" that is used as a penalty term. Higher values of alpha would highly penalize the feature coefficients and the magnitude of the coefficients get reduced.

We tested this model on our data set with a value of alpha set to default 0.05, which gave us an r2_score of 66.17%, which is the same as the one given by linear regression.

Changing the value of alpha to 0.5 and 5 , also did not vary the score by a significant amount, which lead us to believe that our data set does not have multicollinearity problem, and thus using linear models would not be a best fit for our data set.

3.4.3 Decision Tree Regressor

Decision trees build classification/regression models in the form of a tree structure that can be used to predict both categorical and numerical data. In our case, since we predict the number of food borne illnesses, we are trying to predict continuous numerical data.

Decision trees work by breaking down the data sets into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed with the final result as a tree with decision and leaf nodes. The decision node has two or more branches each representing the values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor is used as the root node for the tree.

In case of regression based decision trees , the goal is to find the best predictor as the root node where there is highest amount of standard deviation reduction possible. The algorithm for regression based decision trees is as follows:

Step1: The standard deviation of the target is calculated.
Step2: The data set is split on different independent variables(attributes)
- The standard deviation for each branch is calculated
- The resulting standard deviation is subtracted from the standard deviation before splitting
- The result is the standard deviation reduction
Step3: The attribute with the largest standard deviation reduction is chosen as the decision node for the tree

Using regression based decision tree on our data set with default tree depth of "None" which means that the trees are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples which is set to 2, the gave a training accuracy of 100%, which lead us to believe that the model is trying to overfit the data.

However, testing accuracy as 79.68% was not as high as training accuracy.To avoid over-fitting problem, but looking at accuracy improvement from linear models by 13% we selected Random forest regressor as our next baseline model that leverage multiple decision trees.

3.4.4 Random Forest Regressor

Random Forest Regressors use multiple decision trees and merges them together to provide more stable and accurate predictions.Random forests use a number of classifying decision trees on various sub samples of the data set and uses averaging to improve the predictive accuracy and control the problem of over-fitting as faced by using just decision trees. The sub sample size are the same as the input size but the samples are drawn with replacement.

We used Random forest regressor on our data set that gave a training accuracy of 98.22% and a testing accuracy of 82.03%. This was definitely a great model as compared to decision trees alone since the testing accuracy suggested that the model performed well on unseen data almost efficiently.

3.4.5 Gradient Boosting Regressor

Even though we found Random forest regressor to perform well on our data set,we also went ahead to use gradient boosting regressor and check its accuracy. Gradient Boosting regressor is an ensemble method that uses multiple weak regressor and builds trees sequentially trying to reduce the error at each step. This leads to more over-fitting usually as compared to a Random Forest Model. Gradient Boosting Regressor on our data set performed fairly, with default parameters being said but took a considerably longer to model using training data. The training accuracy for the model was 82.54% and testing accuracy was 78.31%.

4 Model Building

4.1 Task 1: Predict confirmed/suspected case of food borne illness - classification

Based on the results from baseline models, Random Forest Algorithm was chosen to be the best model and taken further for tuning the model hyper parameters to arrive at the best performance.

Hyper parameter tuning was achieved using the concepts of Exhaustive Grid Search. The grid search provided by GridSearchCV exhaustively generates candidates from a grid of parameter values specified with the param_grid parameter. The GridSearchCV instance implements the usual estimator API: when "fitting" it on a data set all the possible combinations of parameter values are evaluated and the best combination is retained[11].

The following figure shows the various parameters that were tuned and the optimal value for the model that would lead to increased testing accuracy achievable by Random Forest Model :

Parameter	Values Tested	Optimal Value Achieved
n_estimators	5,10,15,20	20
max_depth	20,25,30,35,40,45	25
min_samples_split	2,4,6,8,10,20,40,60,100	2
min_samples_leaf	1,3,5,7,9	1

Figure 17: Hyper parameters tuned for Classification Task

The hyper parameters were:
1. n_estimators: The number of trees to be used in the forest.
2. max_depth: The maximum depth of each tree in the forest.
3. min_samples_split: The minimum number of sample required to split an internal node.
4. min_samples_leaf: The minimum number of samples required to be in a leaf node.

4.2 Task 2: Predict number of food borne illness - Regression

After finding the testing accuracy for a number of baseline models detailed above, we decided to go ahead with Random Forest Regressor, since it gives us the best testing accuracy among all the baseline models.
Random forest regressor model was chosen for further fine tuning the hyper parameters with an attempt to improve the accuracy and make it a better at the prediction task.
Since random forest regressor also uses decision trees as the baseline estimators, the hyper parameters tuned here similar to those done for random forest classifier. The figure below shows hyper parameters that were tuned and the optimal value reached for each of these parameters, using GridSearchCV technique:

Parameter	Values Tested	Optimal Value Achieved
n_estimators	20,50,80,100,120,150	120
min_samples_split	2,4,6,8,10,20,40,60,100	2
min_samples_leaf	1,3,5,7,9	1

Figure 18: Hyper parameters tuned for Regression Task

5 Results

5.1 Predict confirmed/suspected case of food borne illness

Figure 19 shows an overview of baseline models.

Model	Test Accuracy	AUC Score
K Nearest Neighbors	59.75%	64.53%
Logistic Regression	63.96%	62.43%
Ada Boost Classifier	66.38%	65.67%
Random Forest Classifier	68.52%	66.52%
Gradient Boosting Classifier	67%	66.66%

Figure 19: Baseline Model results

After using optimal values for the hyper parameters to model and predict for Task1, the model achieved a testing accuracy of 68.7%.The accuracy of the classifier after tuning the parameters increased by just 0.20%.

The following figure shows the model report and the features that played a major role in the prediction task

```
Accuracy of the RF on test set: 0.687
AUC Score: 0.665191
              precision    recall  f1-score   support

           0       0.93      0.36      0.52     16796
           1       0.63      0.98      0.77     19080

   micro avg       0.69      0.69      0.69     35876
   macro avg       0.78      0.67      0.64     35876
weighted avg       0.77      0.69      0.65     35876
```

Figure 20: Model Report after fine tuning hyper parameters

5.2 Task 2: Predict number of food borne illness

Figure 21 gives an overview of baseline models

Model	Training Accuracy	Test Accuracy
Linear Regressor	0.6856	0.6619
Ridge Regressor	0.6856	0.6619
Decision Tree Regressor	1.0000	0.7968
Gradient Boosting Regressor	0.8254	0.7831
Random Forest Regressor	0.9822	0.8203

Figure 21: Baseline Model results

We further tune hyper parameters as stated above and improved testing accuracy by minuscule value.

```
Model Report
Train Accuracy : 0.9891
Test Accuracy : 0.8342
```

Figure 22: Model Report after fine tuning hyper parameters

Hyper parameter tuning in Task 2, increased the testing accuracy of the model by 1.39%.Though this was not an significant improvement over the baseline score, the model seems to overall perform well.

6 Model Evaluation

6.1 Task 1: Predict confirmed/suspected case of food borne illness

For predicting whether a case is suspected/confirmed food borne illness, evaluation metrics like Sensitivity, Specificity and Youden's index have been used. We also checked Area Under the Curve (AUC) values to evaluate the model performance.Sensitivity and Specificity are important metrics that helped evaluate our binary classification model efficiently.

Sensitivity(Recall/True Positive Rate) : Is the proportion of the actual positives that are correctly identified.

$$Sensitivity(TPR) = \frac{TP}{TP + FN}$$

Specificity(True Negative Rate): Measures the proportion of the actual negatives that are correctly identified.

$$Specificity(TNR) = \frac{TN}{TN + FP}$$

From Figure 23, we can see that sensitivity value for this model is as low as 36% when predicting Class'0' and the recall value for predicting Class'1' is 98%. We see that the model has the ability to identify the relevant samples in the data set. Precision also expresses the proportion of the data points the model says as relevant which are actually relevant. As precision value is high, the model discards the False Positives well and since the recall value is also high , the model also discards the False Negatives well.The model has good precision value for Class '0' and also a descent value for Class'1'.Thus overall the model performs well for the dataset.

6.2 Task 2: Predict number of food borne illness

For predicting the number of people to fall ill due to food borne illness, R-Squared metric have been used to evaluate model performance.
The scores that have been presented for the model above are the r2_score values. In case of our random forest regressor model on the data set, the training accuracy was high which is an indication that the model is trying to overfit the data. The model gives good testing score of 83.42% which imply that fitted

```
Accuracy of the RF on test set: 0.687
AUC Score: 0.665191
              precision    recall  f1-score   support

           0       0.93      0.36      0.52     16796
           1       0.63      0.98      0.77     19080

   micro avg       0.69      0.69      0.69     35876
   macro avg       0.78      0.67      0.64     35876
weighted avg       0.77      0.69      0.65     35876
```

Figure 23: Model Report after fine tuning hyper parameters

regression line is such that the data points are close to the line and thus the error is low and therefore the model is able to generalize on new data samples.

7 Conclusion

For both classification and regression tasks, we identified that random forest model best suited our data set. This is because our data set contains both categorical as well as numerical data and tree based models tend to perform better since, trees can accurately divide data based on categorical values well. Tree based model empower the predictive models with high accuracy, stability and ease of interpretation. Since our data set was not collinear, the non-linear relationship mapping is done quite well by the tree based models.

8 Future Work

Though we tried tuning hyper parameters only for random forest models, we also want to tune the gradient boosting models, since they provide a scope for further improvement after tuning some parameters, since the accuracy that we achieved without tuning were very close to random forest models with tuning. We also want to be able to explore the data set in more detail and come up with some more predictions such as, foods that lead to most hospitalizations or fatalities, or combinations of food and their potential source of infection.

References

[1] https://www.kaggle.com/cdc/foodborne-diseases/home

[2] https://web.uri.edu/foodsafety/cause-and-prevention-of-foodborne-illness

[3] https://hackernoon.com/introduction-to-machine-learning-algorithms-logistic-regression-cbdd82d81a36

[4] https://www.kaggle.com/amolbhivarkar/knn-for-classification-using-scikit-learn

[5] https://scikit-learn.org/stable/modules/ensemble.html#adaboost

[6] Ho, Tin Kam (1995). *Random Decision Forests*. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.

[7] Ho TK (1998). *The Random Subspace Method for Constructing Decision Forests*. IEEE Transactions on Pattern Analysis and Machine Intelligence. 20 (8): 832–844. doi:10.1109/34.709601

[8] Hastie, Trevor; Tibshirani, Robert; Friedman, Jerome (2008). *The Elements of Statistical Learning (2nd ed.)*. Springer. ISBN 0-387-95284-5.

[9] https://scikit-learn.org/stable/modules/ensemble.html#random-forests

[10] https://scikit-learn.org/stable/modules/ensemble.html#gradient-tree-boosting

[11] https://scikit-learn.org/stable/modules/grid_search.html

[12] https://en.wikipedia.org/wiki/Multicollinearity

[13] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

[14] https://www.youtube.com/watch?v=nSaOuPCNvlk

[15] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[16] https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/