Title: Study on predicting the retail prices of onion in the local market of Bangladesh.

Author

**Abstract** Demand and supply forces are often expected to determine the prices of a product in the market. However, sometimes these forces may lead to unprecedented price increases in the market, making it necessary to have other forces to help regulate the market prices. In Bangladesh, in October 2021 and March 2022, the demand and supply factors caused a sharp rise in onion prices that led to a public outcry. This price hike was caused by various factors like bad harvesting and restriction on the export of onion by the neighboring country.

In this study, we tried to predict the price of onions using various machine learning models and determine which machine learning model gives a better result.

**Introduction** Onion (*Allium cepa*) is among the most popular vegetables in the world. Onion is a crop that is classified as a cool-season crop. However, it can be grown in a wide range of climatic conditions. It is grown mainly for its bulb, which is used in every home almost daily, across Ethiopia (AgroBIG, [2016](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144)). Ethiopia, the third-biggest producer of onion in the African continent, next to Egypt and South Africa, contributes only 2.7% to the total world production between 2000–2011 (FAOSTAT, [2019](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144)). Averagedthroughoutf 2010 to 2018, the onion area harvested, production, and yield at the national level are 28,942 hectares, 33,947 tons, and 11.70 tons/ha, respectively, (Centrl Statistical Agency (CSA), [2019](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144); Food and Agriculture Organization Statistical Division (FAOSTAT), [2019](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144)), which is far below the world average of 19.7 tons/ha (Megersa, [2017](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144)). In the major rainy season of 2018/19, onion production covered about 11.46% of the country’s root crop area.[1](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144#en0001) In the same season, it was grown by 28,682 smallholder farm households in Tigray, Ethiopia. Together, these households produced 8,223 tons of onion on 1,299.06 hectares with yields of 6.33 tons/ha, which is less than the national average yield of 9.32 tons/ha in the same season (CSA, [2019](https://www.tandfonline.com/doi/full/10.1080/23311932.2020.1712144)).

Onion is one of the major spices crops in Bangladesh. It is used as a common ingredient for its own unique flavor in preparing various types of cooking in our country. It has curative power that makes it essential to use them in the medicinal plant too.

Related works

**Data set** The study area is every subdistrict market of Bangladesh. The data set consists of 7 columns and 6009 rows. The data was collectedfromm ‘www.dam.gov.bd.’

website. The data set includes specific dates and the retail prices of various types of onions in

the subdistrict market. Price consists of 3 values max, min, and mean price. We also added the corresponding Arabic calendar date and USD rate for every date. The ‘MEAN’ column is the output of this dataset and we used every other column to find the relation with the output column. I used leveling the string values ‘location’, ‘type’ ‘date’. As the values of mean column is continues I used the linear regression model to train the machine.

But in order to fit classification models in the data set ,I had to label the values of the ‘MEAN’ column.

**Dataset sample(Before lebeling)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Arabic\_date  **15/5/1442** | Date  2020-12-30 | USD rate  84.78 | Location  Chadpur Sadar | type  Imported | MIN  34.0 | MAX  35.0 | MEAN  34.5 |
| **15/5/1442** | 2020-12-30 | 84.78 | Cox's Bazar Sadar | Imported | 35.0 | 60.0 | 47.5 |
| **15/5/1442** | 2020-12-30 | 84.78 | Khagrachhari Sadar | Imported | 25.0 | 35.0 | 30.0 |
| **15/5/1442** | 2020-12-30 | 84.78 | Lakshmipur Sadar | Imported | 30.0 | 35.0 | 32.5 |
| **15/5/1442** | 2020-12-30 | 84.78 | Rangamati Sadar | Imported | 27.0 | 30.0 | 28.5 |

**Dataset sample(After lebeling)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Arabic\_date | Date | USD rate | Location | type | MIN | MAX | MEAN |
| **9/11/1443** | 1 | 92.92 | 1 | 2 | 50.0 | 60.0 | 2 |
| **9/11/1443** | 1 | 92.92 | 2 | 3 | 25.0 | 30.0 | 3 |
| **9/11/1443** | 1 | 92.92 | 3 | 1 | 34.0 | 35.0 | 4 |
| **9/11/1443** | 1 | 92.92 | 4 | 2 | 30.0 | 35.0 | 5 |

**Supervised Machine Learning** The majority of practical machine learning, uses supervised learning. Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output Y = f(X) . The goal is to approximate the mapping function so well that when you have new input data (x), you can predict the output variables (Y) for that data.

Techniques of Supervised Machine Learning algorithms include linear and logistic regression, multi-class classification, Decision Trees and support vector machines. Supervised learning requires that the data used to train the algorithm is already labelled with correct answers. For example, a classification algorithm will learn to identify animals after being trained on a dataset of images that are properly labelled with the species of the animal and some identifying characteristics.

Supervised learning problems can be further grouped into Regression and Classification problems. Both issues aim to construct a brief model that can predict the value of the dependent attribute from the attribute variables. The difference between the two tasks is that the dependent attributepointical for regression and categorical for classification.

**Classification Problem**

2. Classification Methods Artificial intelligence methods facilitated the classification and the determination of the validity of the results. There were six different ML (supervised learning) methods in this study. These methods are k-nearest neighbors (kNN), support vector machine (SVM), random forest (RF), Decision tree, naïve Bayes (NB), and logistic regression (LR).

2.2.1. k-Nearest Neighbors (kNN) This algorithm makes a clustering process based on the proximity relations between objects. It works in the coordinate plane with the linear decomposition method that obtains neighbor data using the Euclidean distance between data points [14,15].

2.2.2. Support Vector Machine (SVM) finds the best way to classify the data based on the position in relation to a border between positive class and negative class. This border is known as the hyperplane, which maximizes the distance between data points from different classes. Like the decision tree and random forest, a support vector machine can be used in both classification and regression, SVC ; SVCort vector classifier) is for classification problems.

2.2.3. Random Forest (RF) random forest is a collection of decision trees. It is a common type of ensemble method which aggregates results from multiple predictors. Random forest additionally utilizes a bagging technique that allows each tree to be trained on a random sampling of the original dataset and takes the majority vote from trees. Comparthe ed to decision tree, it has better generalization but is less intrpretable, because of more layers added to the model.

2.2.4. Decision tree builds tree branches in a hierarchical approach, and each branch can be considered an if-else statement. The branches develop by partitioning the dataset into subsets based on the most imessentialeatures. Final classification happens at the leaves of the decision tree.

2.2.5. Naïve Bayes (NB) The naïve Bayes re-scans the entire dataset for each new classification operation which might cause it to operate relatively slowly.

2.2.6. Logistic Regression (LR) LR is the iterative presentation of the powerful linear combination of variables most likely to determine the observed outcome.

**Code for fitting the data frame to classification models**

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.naive\_bayes import GaussianNB

model\_pipeline = []

model\_pipeline.append(LogisticRegression(solver='liblinear'))

model\_pipeline.append(SVC())

model\_pipeline.append(KNeighborsClassifier())

model\_pipeline.append(DecisionTreeClassifier())

model\_pipeline.append(RandomForestClassifier())

model\_pipeline.append(GaussianNB())

from sklearn import metrics

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report

from sklearn.metrics import confusion\_matrix

model\_list = ['Logistic Regression', 'SVM', 'KNN', 'Decison Tree', 'Random Forest', 'Naive Bayes']

acc\_list = []

cm\_list = []

for model in model\_pipeline:

  model.fit(X\_train, y\_train)

  y\_pred = model.predict(X\_test)

  acc\_list.append(metrics.accuracy\_score(y\_test, y\_pred))

  cm\_list.append(confusion\_matrix(y\_test, y\_pred))

import numpy as np

result\_df = pd.DataFrame({'Model':model\_list, 'Accuracy': acc\_list})

#result\_df=result\_df.round({"Accuracy":2})

result\_df['Accuracy'] =  np.round(result\_df['Accuracy'], decimals = 3)

#result\_df.round(2)

print(result\_df)

**Code for getting Sensitivity(sn), specifity(sp), F1 Score and MCC**

# Creating a function to report confusion metrics

def confusion\_metrics (conf\_matrix):

# save confusion matrix and slice into four pieces

    TP = conf\_matrix[1][1]

    TN = conf\_matrix[0][0]

    FP = conf\_matrix[0][1]

    FN = conf\_matrix[1][0]

    print('True Positives:', TP)

    print('True Negatives:', TN)

    print('False Positives:', FP)

    print('False Negatives:', FN)

    # calculate accuracy

    conf\_accuracy = (float (TP+TN) / float(TP + TN + FP + FN))

    # calculate mis-classification

    conf\_misclassification = 1- conf\_accuracy

    # calculate the sensitivity

    conf\_sensitivity = (TP / float(TP + FN))

    # calculate the specificity

    conf\_specificity = (TN / float(TN + FP))

    # calculate precision

    conf\_precision = (TN / float(TN + FP))

    # calculate f\_1 score

    conf\_f1 = 2 \* ((conf\_precision \* conf\_sensitivity) / (conf\_precision + conf\_sensitivity))

    print('-'\*50)

    print(f'Accuracy: {round(conf\_accuracy,2)}')

    print(f'Mis-Classification: {round(conf\_misclassification,2)}')

    print(f'Sensitivity: {round(conf\_sensitivity,2)}')

    print(f'Specificity: {round(conf\_specificity,2)}')

    print(f'Precision: {round(conf\_precision,2)}')

    print(f'f\_1 Score: {round(conf\_f1,2)}')

for cm in cm\_list:

  confusion\_metrics (cm)

**Table: without feature selection**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | Sn /recall | sp | acc | MCC | F1 |
| Logistic Regression (LR) | undefined | 1 | 21% | 0 | undefined |
| Support Vector Machine (SVM) | undefined | 1 | 32% | 0 | undefined |
| k-Nearest Neighbors (KNN) | 1 | 1 | 50% | 0 | 1 |
| Decision Tree | 1 | 1 | 97% | 0 | 1 |
| Random Forest(RF) | 1 | 1 | 92% | 0 | 1 |
| Naive Bayes(NB) | 1 | 1 | 75% | 0 | 1 |

**Top 3 classifiers with feature selection**

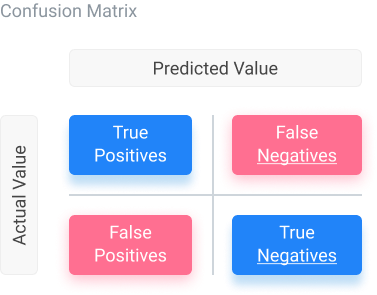
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Classifier | Number of Features | Sn /recall | sp | acc | MCC | F1 |
| Logistic Regression (LR) | 4 | 1 | 1 | 14 | 0 | 1 |
| Support Vector Machine (SVM) | 4 | undefined | 1 | 28 | 0 | undefined |
| k-Nearest Neighbors (KNN) | 4 | 1 | 1 | 36 | 0 | 1 |
| Decision Tree | 4 | undefined |  | 56 | 0 | undefined |
| Random Forest(RF) | 4 | 1 | 1 | 58 | 0 | 1 |
| Naive Bayes(NB) | 4 | undefined | 1 | 48 | 0 | undefined |

# Result

### **Confusion Matrix to evaluate model performance**

A confusion matrix is used to display parameters in a matrix format. It allows us to visualize true and false positives and true and false negatives.

To get the overall accuracy, we can subtract total false positives and negatives from the total number of tests and divide that by the total number of tests. We can use a confusion matrix by importing it through the *sklearn* library. Scikit-learn (sklearn) in Python contains tools for machine learning and statistical modeling, including methods from [classification to dimensionality reduction](https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/).

Confusion matrix in ML is used for evaluating the precision of a classification model.

The following lines of code would import and implement a confusion matrix, with the assumption that *y\_pred* and *y\_test* have been initialized previously. In the following Python example, y\_test and y\_pred are variables that represent the tested and predicted values outputted by the machine learning model.

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

print (cm)

## **What Is Sensitivity (sn)**

Sensitivity is a measure of the proportion of actual positive cases that got predicted as positive (or true positive). Sensitivity is also termed as Recall. This implies that there will be another proportion of actual positive cases, which would get predicted incorrectly as negative (and, thus, could also be termed as the false negative). This can also be represented in the form of a false negative rate. The sum of sensitivity and false negative rate would be 1. Let's try and understand this with the model used for predicting whether a person is suffering from the disease. Sensitivity is a measure of the proportion of people suffering from the disease who got predicted correctly as the ones suffering from the disease. In other words, the person who is unhealthy actually got predicted as unhealthy.

Mathematically, sensitivity can be calculated as the following:

Sensitivity = (True Positive)/(True Positive + False Negative)

## **What Is Specificity? (sp)**

Specificity is defined as the proportion of actual negatives, which got predicted as the negative (or true negative). This implies that there will be another proportion of actual negative, which got predicted as positive and could be termed as false positives. This proportion could also be called a false positive rate. The sum of specificity and false positive rate would always be 1. Let's try and understand this with the model used for predicting whether a person is suffering from the disease. Specificity is a measure of the proportion of people not suffering from the disease who got predicted correctly as the ones who are not suffering from the disease. In other words, the person who is healthy actually got predicted as healthy is specificity.

Mathematically, specificity can be calculated as the following:

Specificity = (True Negative)/(True Negative + False Positive)

**Accuracy**

Accuracy is one metric for evaluating classification models. Informally, **accuracy** is the fraction of predictions our model got right. Formally, accuracy has the following definition:

Accuracy=Number of correct predictions/Total number of predictions

For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

Accuracy=(TP+TN) / (TP+TN+FP+FN)

Where *TP* = True Positives, *TN* = True Negatives, *FP* = False Positives, and *FN* = False Negatives.

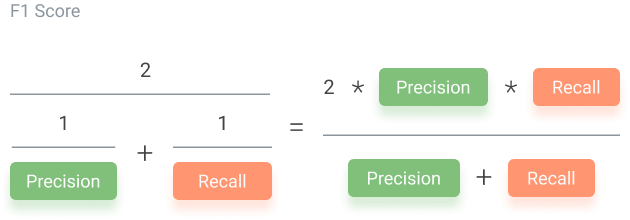
### **Specificity of Machine Learning Models**

Specificity entails the percentage of negative instances out of the total actual negative instances. It is similar to the true positive rate method above. Here, the denominator is the sum of real numbers not generated by the model but instead verified by data. Because the numerator is the number of true negatives, this equation allows us to see how often the model is correct when it generates negative outputs based on the true negatives of the total negatives it produced in the past.

### **F1 Score to analyze Machine Learning Models**

F1 score is when we take the mean of precision and recall. It takes the contribution of both of them, which means that the higher the F1 score, the more accurate the model. Conversely, if the product in the numerator dips down too low, the final F1 score decreases dramatically.

A model with a good F1 score has the most drastic ratio of “true:false” positives as well as the most drastic “true:false” negatives ratio. For example, if the number of true positives to the number of false positives is 100:1, that will play a role in producing a good F1 score. Meanwhile, having a close ratio, say 50:51 true to false positives, will produce a low F1 score. The equation for the F1 score is below.

F1 Score is the weighted average of Precision and Recall.

An F1 score is considered perfect when it’s 1, while the model is a total failure when it’s 0. Thus, a low F1 score is an indication of both poor precision and poor recall.

**Key Terms** Onion, Decision Tree, Random Forest, K-Nearest Neighbor, Logistic Regression, Classification, Regression.

**Conclusion** Onions comprise an important component of the Indian diet. Their demand has increased considerably in the recent past, and the growing demand has been met by domestic production. Yet, the onion prices have risen and become more volatile, shooting up periodically. In this paper, I have tried to demystify the causes of volatility in onion prices