Creating a Machine Learning Algorithm to Classify Raisins

[redacted] *Student, Durham University*

1 SUMMARY

RAISINS are the most common dried fruit in the world, with the drying process removing the water from the grapes, providing a delicious, energy dense snack and cooking ingredient. (Bjarnadottir, 2023) [1]

The human element of analysing food tends to introduce a lack of speed and the potential of mistakes arising. Depending on a human's psychological and physiological condition, their performance at food identification may worsen, therefore the need to look for other methods of culinary analysis becomes apparent. Machine learning and computer vision is one of these 'other methods', which automates the process and removes the potential for humans to make mistakes. With computer vision, features can be extracted from the images. These features can then be fed into a machine learning algorithm and various algorithms can be used to determine a suitable output. (Mollazade, 2012; Omid, 2012; Arefi, 2012) [2]

The dataset used is from Cinar, Koklu and Tasdemir's work, found on the UCI Machine Learning Archive, with 900 instances and 7 morphological features, characterising each raisin grain (Cinar, 2020; Koklu, 2020; Tasdemir, 2020), and without them, this paper would not be possible. [3] Those morphological features are as follows: area, major axis length, minor axis length, eccentricity, convex area, extent and perimeter. Based on these features, the machine learning algorithm classifies an instance into being either a Kecimen or Besni raisin, as there are some differences between the raisins cultivated in these two Turkish towns.

The input data is an Excel spreadsheet, which is read and transformed into a Pandas DataFrame. (Pandas development team, 2024) [4]. The features of the data differ in range, with Area and Convex Area being in the range of [5000, 25000], the Eccentricity and Extent in the range [0, 1], Major and Minor axis lengths being in the range [50, 1000] and the perimeter being [100, 2800]. The target variable is the class of raisin (i.e. Kecimen or Besni). The data will be split into 80 percent for training and 20 percent for testing the model. The goal is to build a predictive machine learning model that takes various pieces of data from processed images of raisins and classify them into Kecimen and Besni raisins.

Figure 1 shows a proposed diagram of the final machine learning solution, showing how the data will be preprocessed via principal component analysis before being fed into the logistic regression algorithm. The data will be split 80/20 between the training and test set to ensure a large enough training set to make the algorithm as robust as possible, with a large enough test set to ensure the algorithm does not overfit to the training set.

The ideal, most accurate end system ended up being a logistic regression model with principal component analysis, giving a roughly 93.8% accurate model based on the test set given to the model after training it. This acted as a valuable personal lesson into dimensionality reduction and weighing up the benefits and potential issues with various classification models. The limitations of the model are that the data has to be in a very specific format, since the original group of researchers used their own image processing on the raisin images in order to gain measurements in pixels. A similar algorithm would need to be used before the data is then passed into the machine learning algorithm.

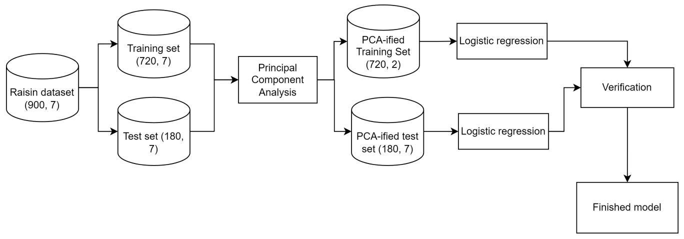


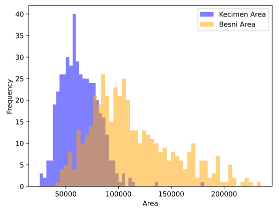
Fig. 1: Diagram illustrating the proposed machine learning system

2 DATA AND EXPERIMENTAL SETUP

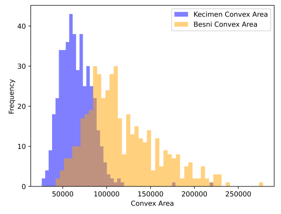
The first step to building any sort of good machine learning algorithm is understanding the data. In this instance, what specifically distinguishes a Kecimen raisin from one of the Besni variety? By comparing the various features of the two raisin types via the utilisation of graphs and other data analysis methods, it becomes possible to visualise the differences between the two, and to have the ability to consider which algorithms and methods would be more suitable for the task.

Figures 2a-2g are all histograms comparing assorted factors of Kecimen and Besni raisins with one another, with Kecimen in blue and Besni in orange.

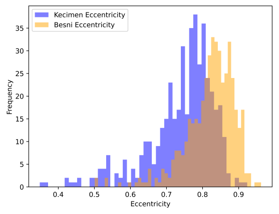
It quickly becomes apparent that, while the two appear similar, the figures illustrate that there are some key differences. The major and minor axis lengths of both raisin varieties have some overlap, but Besni raisins tend to have the larger axis lengths, with more spread compared to the



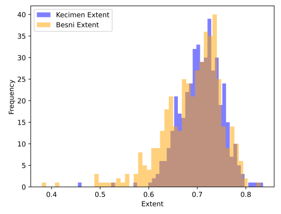
(a) Area of raisins



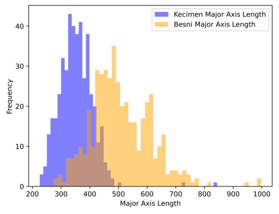
(b) Convex area of raisins



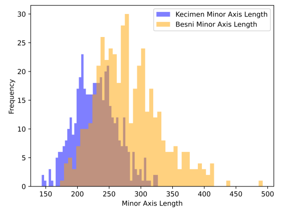
(c) Eccentricity of raisins



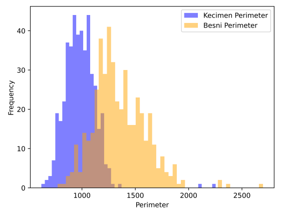
(d) Extent of raisins



(e) Major axis length of raisins



(f) Minor axis length of raisins



(g) Perimeter of raisins

Fig. 2: Features of both raisin types, compared

Feature	Raisin Type	Mean	Median	IQR
Area	Kecimen	63413.467	61420.0	24686.75
	Besni	112194.789	104426.5	52084.75
Convex Area	Kecimen	65696.356	63826.5	25389.25
	Besni	116675.824	108062.5	52111.75
Eccentricity	Kecimen	0.742	0.766	0.098
	Besni	0.821	0.832	0.081
Extent	Kecimen	0.708	0.71	0.057
	Besni	0.691	0.703	0.077
Major Axis Length	Kecimen	352.859	350.234	76.826
	Besni	509.001	493.186	145.841
Minor Axis Length	Kecimen	229.353	228.62	45.588
	Besni	279.624	273.367	65.666
Perimeter	Kecimen	983.686	977.93	188.074
	Besni	1348.127	1305.8	328.137

TABLE 1: Comparing different mean, median and interquartile range of features of Kecimen and Besni raisins

Kecimen raisins. Generally speaking, the Besni raisins have a wider range and slightly higher mean value, but there are some overlaps between the two raisin types.

After looking at the data in Table 1, the Besni raisins tend to be larger in size. The measurements are in pixels, as the original dataset from Cinar, Koklu and Tasdemir involved taking images of the raisins and putting them through image processing to get measurements of each of the features. Table 1 does support that Besni raisins tend to have a higher mean in the majority of features, except for Extent, in which they are essentially level with Kecimen raisins. The table does also support that the interquartile range of Besni raisins is larger.

In order to transform the data, the project made use of scikit-learn's built-in StandardScaler, which calculates the standard score for each feature variable, calculated as $z = (x - u)/s$, where u is the mean of the samples and s is the standard deviation of the samples. By scaling the numbers, it transforms each of the features into having a normal distribution, with 99.7% of the scaled features falling within 3 standard deviations of the mean.

The model will be evaluated using 3 different methods: a confusion matrix, a classification report and an accuracy score, all of which are built-in features with scikit-learn. (Pedgerosa et. al., 2011) [5] The confusion matrix evaluates the predictions at the end by putting all types of classification into a $2 * 2$ matrix. The matrix is in the format of

$$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$$

where TP represents a true positive, TN is a true negative, FP is a false positive and FN is a false negative. In this instance, Besni is considered 'positive' and Kecimen is considered 'negative', as it is a classification task. A false positive or negative in this instance relates to the algorithm misclassifying the raisin. That is, to say, a Besni raisin is classified by the algorithm as a Kecimen raisin, or vice-versa.

The classification report from scikit-learn shows precision, recall and F1 score of the prediction result. The precision is defined as $\frac{TP}{TP+FP}$, which measures how many of the positive predictions, or, in this instance, raisins classified as Besni, are correct. Recall, on the other hand, is defined as $\frac{TP}{TP+FN}$, which is how many positive predictions are correct over all of the positive classifications made. The F1-score takes both the precision and recall metrics and calculating the harmonic mean of the two. It is considered more accurate than just a regular mean, as it is more sensitive to inputs with lower values. The F1-score is calculated as $2 * \frac{P*R}{P+R}$, where P is precision and R is recall. [6].

The accuracy score is the most simple part of this scoring, giving a final 'percentage score', so to speak, to analyse how accurate the prediction model is. The calculation for accuracy is $\frac{TP+TN}{TP+FP+FN+TN} * 100$, which is the total number of correct predictions made divided by the number of predictions overall, then multiplied by 100 at the end to give a percentage score.

3 Model Evaluation

As this classification task is binary, with there only being two classes to classify into, the level of complexity within model selection is reduced. One of the challenges in selecting good models was one that could easily differentiate the two raisin varieties apart, since there was a lot of crossover in some of the graphs seen in Figure 2, namely the raisin extent highlighted in Figure 2d, or the eccentricity from Figure 2c.

After some careful consideration, the two models selected were PCA (principal component analysis) + Logistic Regression, as the baseline and a Random Forest Classifier as the proposed solution. The random forest classification technique seemed to be the most sound, as it was less sensitive to outliers within the data due to the majority voting of the multiple random trees used. Random forest classification would also be good at identifying more complex relationships between different features within the raisins, as logistic regression assumes a linear relationship, or otherwise, between the variables.

Initially, scikit-learn's LogisticRegression was scoring relatively highly without doing PCA, but this was purely experimentation before having explored the data. Following some careful data analysis and some brief thinking, the conclusion was to use principal component analysis, a dimensionality reduction technique that aims to extrapolate the key features from a given dataset for easier visualisation and to reduce the number of random variables. By reducing the data in this way, it prevents the machine learning algorithm from overfitting as the feature set is transformed into the principal components. Due to the overlap shown in Figures 2c and 2d, it becomes an effective method of noise reduction, as the excessive overlap in eccentricity and extent may cause some issues.

Scikit-learn uses random states in some of the procedures, such as the procedure that splits into train and test sets of features, or the random forest classifier. To ensure reproducability of results and an accurate report, the random state has been set to 314159 throughout the program, wherever an element of randomness may be present.

For the original logistic regression, with no principal component analysis, its confusion matrix was

$$\begin{bmatrix} 69 & 6 \\ 12 & 93 \end{bmatrix}$$

with a precision and recall score of 0.85 and 0.92 respectively for Besni; 0.94 and 0.89 respectively for Kecimen. Besni's F1-score was 0.88 and Kecimen's F1-score was 0.91, giving it an accuracy score of 90%.

The training process for this algorithm was merely fitting the original logistic regression model to a scaled version of the feature set, then testing it against the target test set and using the evaluation methods to get the above results.

Following on from this, in an attempt to garner a more accurate regression model, the use of scikit-learn's principal component analysis algorithm was employed to extract the most important information from the data and to ensure that the previously aforementioned overlap illustrated in Figures 1c and 1d were not causing any sort of issues.

Measurement	Raisin Type	LR	LR + PCA	RFC
Precision	Besni	0.85	0.91	0.89
	Kecimen	0.92	0.96	0.93
Recall	Besni	0.92	0.95	0.91
	Kecimen	0.89	0.93	0.92
F1-score	Besni	0.88	0.93	0.90
	Kecimen	0.91	0.95	0.93
Accuracy (%)		90	93.89	91.67

TABLE 2: Comparing different mean, median and interquartile range of features of Kecimen and Besni raisins

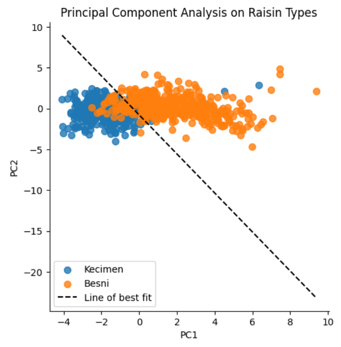


Fig. 3: Principal component analysis on raisin types

The confusion matrix of the new logistic regression model with principal component analysis was

$$\begin{bmatrix} 71 & 4 \\ 7 & 98 \end{bmatrix}$$

with a precision and recall score of 0.91 and 0.95 respectively for Besni; 0.96 and 0.93 respectively for Kecimen. Besni's F1-score was 0.93 and Kecimen's F1-score was 0.95, giving this model an accuracy score of 93.89%, an increase of roughly 3.8% from the original logistic regression model without using principal component analysis.

Next, a random forest classifier was used. As mentioned before, the same random state number was used to ensure replicability of results and report accuracy. This random forest classifier was chosen to capture a potentially non-linear relationship between Kecimen and Besni raisins, as well as ignoring outliers due to the result taking a 'majority vote' from multiple decision trees. The confusion matrix of this random forest classifier model was

$$\begin{bmatrix} 68 & 7 \\ 8 & 97 \end{bmatrix}$$

with a precision and recall score of 0.89 and 0.91 respectively for Besni; 0.93 and 0.92 respectively for Kecimen. Besni's F1-score was 0.90 and Kecimen's F1-score was 0.93, giving this model an accuracy score of 91.67%, a 1.6% increase in accuracy from the original logistic regression model without the use of principal component analysis, but a 2.2% decrease in accuracy compared to the logistic regression model using principal component analysis.

These results are summarised nicely in Table 2.

A key part within machine learning is hyperparameter tuning. This is a method within developing a machine learning algorithm to control model capacity and to control the training process. It ensures that there is minimal, but preferably zero, overfitting within the data, and differs between each machine learning task. The approach I decided to take was to look into scikit-learn's GridSearchCV method, which tries different combinations of hyperparameters in order to spot check combinations known to form well.

Initially, this approach was used with the logistic regression + PCA task, but after going through each hyperparameter that was known to work well for the task at hand, the results shown in Table 2 were exactly the same after the hyperparameter tuning as they were before. The measurement being used here was scikit-learn's accuracy score, which was explained earlier in the report. This accuracy result before and after doing the hyperparameter tuning were identical, with 93.89% accuracy. Since this had made no conceivable difference, and using the same process on the random forest classification would take a considerable length of time, the decision to leave out hyperparameter tuning seemed the most sensible for this task, as it would have taken valuable time that would have most likely been wasted, considering its effects, or lack thereof, on the logistic regression algorithm.

Scikit-learn uses default values for hyperparameter variables within the different models, ranging from regularisation strength and penalty for logistic regression to split quality measurements (criterion) and the number of features to consider when looking for the best split for random forest classification; these default values tend to work quite well for basic machine learning tasks. Since the dataset that is being worked with in this instance only has 900 instances and 7 morphological features, using the default hyperparameters provided by scikit-learn seemed to be the most logical conclusion.

4 Self Evaluation Report

[redacted] My original dataset was incredibly large, with over 32,000 instances and 57 features. While this would have made for an interesting task, the realisation that my own abilities may have been overestimated had begun to come to fruition. This prompted finding another dataset, which was considerably smaller, having 900 instances and 7 morphological features, as mentioned earlier, and easier to work with on a personal level. Finding this balance between small enough to be manageable for myself and large enough to create a good machine learning model was a difficult task, but my general belief is that this has been accomplished to an acceptable level.

[redacted]

[redacted]

If given the chance to do the coursework again, one of the first priorities would be to select a good dataset the first time around and giving myself the ability to have enough time to turn the project around without needing to apply for an extension. As well as this, watching extra content around machine learning, such as the ”Machine Learning for Everybody” video on YouTube from freeCodeCamp.org, may have helped to gain additional perspectives on content that had been previously covered in the lecture slides, and may have resulted in a deeper, more well-rounded understanding of the concepts.

Unfortuantely, no novel methods or unique approaches were taken regarding this machine learning task. All of what has been done in this project comes from piecing together various materials found online, namely in the scikit-learn documentation, to create a version of a machine learning task to classify raisins, based on the work of the aforementioned Cinar, Koklu and Tasdemir, with the goal being to make this machine learning algorithm more accurate than theirs, rather than attempting to ”reinvent the wheel”, so to speak.

References

[1] BJARNADOTTIR, A., (2023, July), Dried Fruit: Good Or Bad?, Healthline, https://www.healthline.com/nutrition/dried-fruit-good-or-bad

[2] MOLLAZADE, K., OMID, M., AREFI, A., (2012, June), Comparing data mining classifiers for grading raisins based on visual features, Computers electronics in agriculture, vol. 84, pp. 124-131 https://www.sciencedirect.com/science/article/pii/S016816991200066X

[3] CINAR I., KOKLU M. and TASDEMIR S., (2020, December), Classification of Raisin Grains Using Machine Vision and Artificial Intelligence Methods. Gazi Journal of Engineering Sciences, vol. 6, no. 3, pp. 200-209

[4] The pandas development team. (2024, January). pandas-dev/pandas: Pandas (v2.2.0). Zenodo. https://doi.org/10.5281/zenodo.10537285

[5] PEDREGO, F., VAROQUAUX, GA”EL, GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., et. al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, vol. 12(Oct), p.2825–2830.

[6] KANSTRÉN, T., (2020, September), A Look at Precision, Recall and F1-score, towardsdatascience.com, https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec