

1 Data Mining: BeeViva Challenges

2
3 MARCO BELLÒ, University of Padua, Italy
4

5 All code, images, L^AT_EX sources, as well as a less polished version of this report in markdown format, split into two files for the two
6 challenges, can be found at the following repository: <https://github.com/mhetacc/DataMiningChallenges/>.
7

8 **ACM Reference Format:**

9 Marco Bellò. 2025. Data Mining: BeeViva Challenges. 1, 1 (December 2025), 5 pages. <https://doi.org/XXXXXXX.XXXXXXX>
10

11 **1 RICE VARIETIES**

12 **1.1 Preliminary Observations**

13 *1.1.1 Dataset.* All following considerations are made using the datasets provided on the challenge page (*rice_test.csv*
14 and *rice_train.csv*), but it is worth noting that they both stem from an original one that can be found at the [dataset](#)
15 [source page](#). I used it to compute an alternative version of the test dataset that contains the true attribute for *Class*
16 (listing 1).

20
21 Listing 1. Merge datasets

```
22 1 df1 = pd.read_csv("rice_test.csv").round(10)
23 2 df2 = pd.read_csv("Rice_Cammeo_Osmancik.csv").round(10)
24 3
25 4 keys = [col for col in df1.columns]
26 5 merged = df1.merge(df2, on=keys, how="inner")
```

27
28 *1.1.2 Scatter Plot.* From the *scatter plot* (figure 1) we can infer that there are four features (*Area*, *Perimeter*, *Convex_Area*
29 and *Major_Axis_Length*) that seem to be extremely correlated
30

31 *1.1.3 Correlation Matrix.* We can use a correlation matrix (table 1) to see exactly how related to each other the features
32 are. As suspected, all four features before-mentioned have a high degree of correlation (over ninety percent). Once way
33 to handle this is to either use models robust against collinearity, or to either combine or remove some of the correlated
34 features.
35

36 *1.1.4 Features' Importance.* To get a better idea on the importance of each feature, I trained a simple linear regression
37 model, and looked at the result with *summary(fit)*, which can be seen in table 2. The significance stars tells us visually
38 that, with the exception of *Minor_Axis_Length*, the features that are highly correlated to each other contributes strongly
39 to the model, while the *Eccentricity* and *Extent* contribute very little.
40

41 Author's address: Marco Bellò, marco.bello.3@studenti.unipd.it, University of Padua, Padua, Italy.
42

43 Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not
44 made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components
45 of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on
46 servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
47

48 © 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
49

50 Manuscript submitted to ACM
51

52 Manuscript submitted to ACM

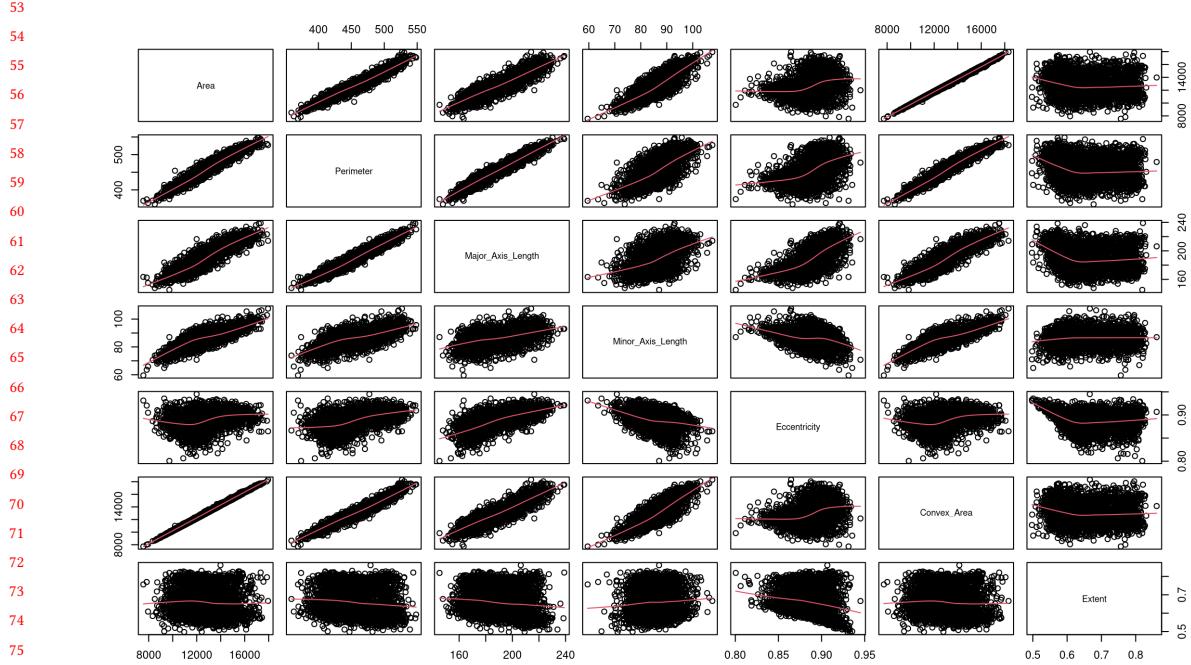


Fig. 1. Scatter Plot for the *rice_train* dataset with LOESS smoothing lines for each class.

Table 1. Correlation matrix of features

	Area	Perimeter	Major Axis Length	Minor Axis Length	Eccentricity	Convex Area	Extent
Area	1.00000000	0.96704011	0.90356325	0.79022701	0.34653177	0.99895272	-0.06002223
Perimeter	0.96704011	1.00000000	0.97163180	0.63486482	0.53784650	0.97046788	-0.12718015
Major Axis Length	0.90356325	0.97163180	1.00000000	0.45675159	0.70625660	0.90390912	-0.13562592
Minor Axis Length	0.79022701	0.63486482	0.45675159	1.00000000	-0.29393931	0.78975480	0.06192558
Eccentricity	0.34653177	0.53784650	0.70625660	-0.29393931	1.00000000	0.34707340	-0.19301157
Convex Area	0.99895272	0.97046788	0.90390912	0.78975480	0.34707340	1.00000000	-0.06397213
Extent	-0.06002223	-0.12718015	-0.13562592	0.06192558	-0.19301157	-0.06397213	1.00000000

We can also measure the total variance explained by all predictors combined using $R^2 = 0.6953849$, and we can check the standardized coefficients to better compare predictors' importance, as shown in table 3. Once again, we can see that Extent and Eccentricity contribute very little to the overall model.

1.1.5 Principal Components. A good way to aggregate and simplify data is by decomposing it into its principal components (centered in zero and scaled to have unit variance). Components' summary can be seen in table 4, and it shows us that more than ninety nine percent of overall variance can be explained with just three components, so a 2D visualization with only the first two components should give us a good idea of the whole dataset (figure 2). The plot suggests that data is well separated, with similar classes clustering nicely. This should make classification easier.

Table 2. Regression coefficients and coefficients' importance

Parameter	Estimate	Std. Error	t value	Pr(> t)	Importance
(Intercept)	-2.152e+00	1.990e+00	-1.081	0.279633	
Area	5.601e-04	1.023e-04	5.474	4.79e-08	***
Perimeter	8.440e-03	2.221e-03	3.800	0.000148	***
Major_Axis_Length	-2.198e-02	5.709e-03	-3.851	0.000120	***
Minor_Axis_Length	4.669e-02	1.213e-02	3.850	0.000121	***
Eccentricity	4.534e+00	1.828e+00	2.481	0.013170	*
Convex_Area	-8.609e-04	9.418e-05	-9.141	< 2e-16	***
Extent	7.146e-02	7.144e-02	1.000	0.317237	

Table 3. Standardized regression coefficients

Parameter	Standardized Coefficient
(Intercept)	NA
Area	1.95970669
Perimeter	0.60605955
Major_Axis_Length	-0.77272038
Minor_Axis_Length	0.54256082
Eccentricity	0.18931314
Convex_Area	-3.09099064
Extent	0.01114328

Table 4. Summary of principal components

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
Standard deviation	2.2924	1.2436	0.9562	0.51393	0.10621	0.07758	0.04519	0.02052
Proportion of Variance	0.6569	0.1933	0.1143	0.03302	0.00141	0.00075	0.00026	0.00005
Cumulative Proportion	0.6569	0.8502	0.9645	0.99753	0.99894	0.99969	0.99995	1.00000

1.2 Modeling

After exploring the dataset, I tried to find the best performing model for predicting the target.

1.2.1 Linear Regression. First I split the training dataset into training and validation sets, fitted the model and compute its *MSE*. From this basic regression I got $MSE = 0.07354557$ and $RMSE = 0.2711929$. Since we know from the preliminary observations that some features are highly correlated (so the dataset suffers from collinearity), I tried to transform the dataset to reduce it. A straightforward solution is to compute the average of all of them (listing 2), but this yield a slightly worse result: $MSE \approx 0.074$.

Another approach would be to drop features altogether, but first we want to measure their variance inflation factor. As a rule of thumb, a *VIF* value above then indicates a problematic amount of collinearity, so I tired to drop features iteratively to see if the model improves, as shown in table 5. For each dropped variable I computed the *VIFs* and the *MSE* of the resulting model. As we can see, dropping *Area* resulted in performances similar to combining the variables, while dropping the other features worsened prediction performances even further.

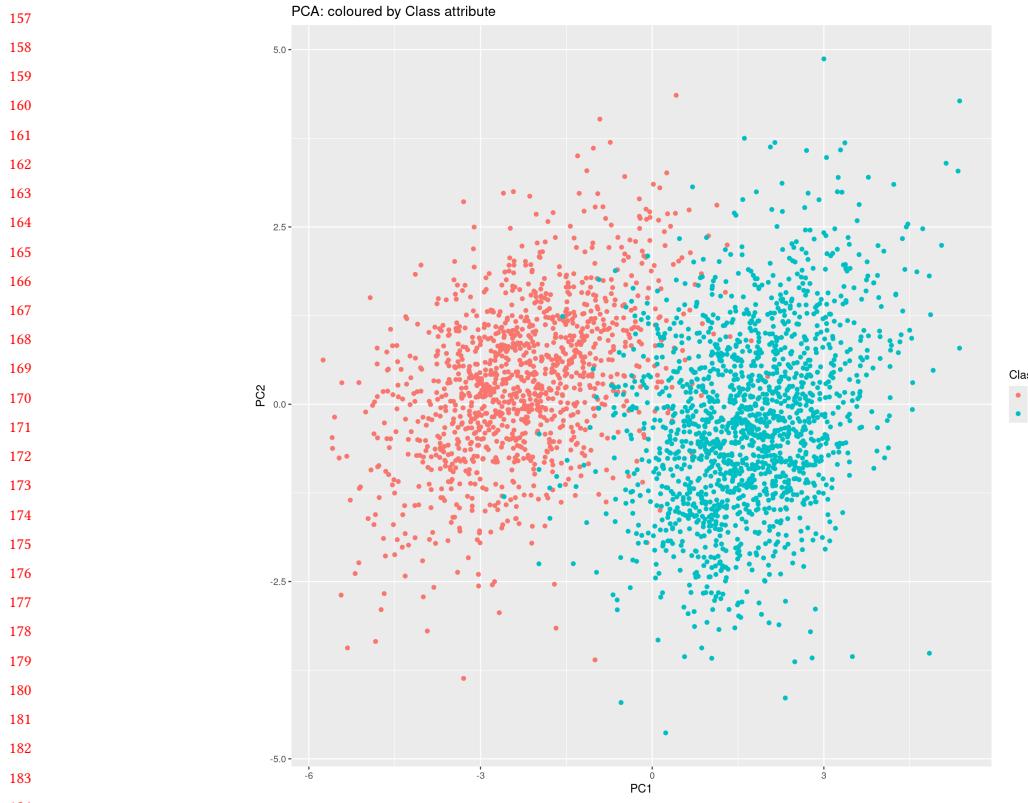


Fig. 2. Plot with first two principal components of *rice_train* dataset.

Table 5. Effect of variable removal on variance inflation factors (VIF) and mean squared error (MSE)

Variable Dropped	VIF Perimeter	VIF Major_A_L	VIF Minor_A_L	VIF Ecc.	VIF Convex_A	VIF Extent	MSE
Area	114	206	133	53	332	1	0.0735629
Convex Area	108	134	40	49	-	1	0.07572877
Major Axis Length	47	-	37	29	-	1	0.07718435
Perimeter	-	-	1	1	-	1	0.0806868

Listing 2. Average of highly correlated features

```
1 rice.train$Combined <- rowMeans(rice.train[, c("Area", "Perimeter", "Major_Axis_Length", "Convex_Area")])
2
3 linear_fit = lm(Class ~
4   Minor_Axis_Length
5   + Eccentricity
6   + Extent
7   + Combined,
8   data=rice_train,
9   subset = train
10 )
11
```

```
209 12 pred <- predict(linear_fit, newdata = rice_train[test, ])
210 13 mean((pred - y.test)^2)      # [1] 0.07439239
211
212 1.2.2 Ridge Regression.
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
```