Report for Assignment 2

by XINGLONG LI – 21115109

The forest fires dataset has twelve columns for the features and one ('Area') for the target, the wine quality dataset has thirteen columns for the features and one (‘quality) for the target and the abalone dataset has eight columns for features and one (‘Rings’) for the target. According to the official description of the datasets, there are no missing value in each column (but we still need to guarantee the point by exploring the real dataset). Our aim is to use different representation learning algorithm and compare the regression performance (RMSE) of different models, including the KNN Regressor, Random Forest Regressor and Gradient Boosting Regressor.

This report builds on the previous report from Assignment 1. Therefore, in the Feature Engineering section, we focus solely on the Forest Fires dataset and do not provide a detailed discussion of other datasets. Additionally, we provide necessary mathematical derivations to support our analysis of the relationship between experimental results and the principles of the models. These derivations are provided in the Supplemental section. **The derivation of MLE and MAP is placed AFTER the Supplemental section** as a standalone part, independent of the main report.

To analyze models rigorously, we employ linear algebra. We represent the input data points as a real-valued matrix A of size m by n, where each column vector corresponds to a data point after centering.

1. Target values transform

In forest fires dataset, the distribution plot of 'Area' (Figure 1) shows that this output variable is very skewed towards 0.0 and varies over an extremely wide range (from 0 to 1090.84), so we first transform it with a ln(x+1) logarithm function and then 'recover' it with an e exponential function e^x - 1 when models predict on the test dataset.

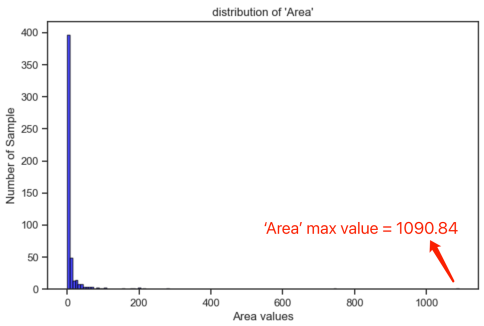

Figure 1. frequency plot of ‘Area’

2. Feature Engineering

2.1. Forest fire dataset

2.1.1. Discrete Feature Encoding

We need to check if categorical and integer features with limited discrete values require encoding. The features 'month', 'day', 'X', and 'Y' (see frequency plots in Figure 2) are discrete and should be one-hot encoded, as their values are independent with no numerical relationship (e.g., $0 < 1 < 2$ or $6 = 2 \times 3$ does not apply). For instance, locations with $X = 2$ and $X = 8$ are unrelated in magnitude since coordinates lack numerical meaning. Additionally, their domain sizes (<13) are small, making one-hot encoding appropriate.

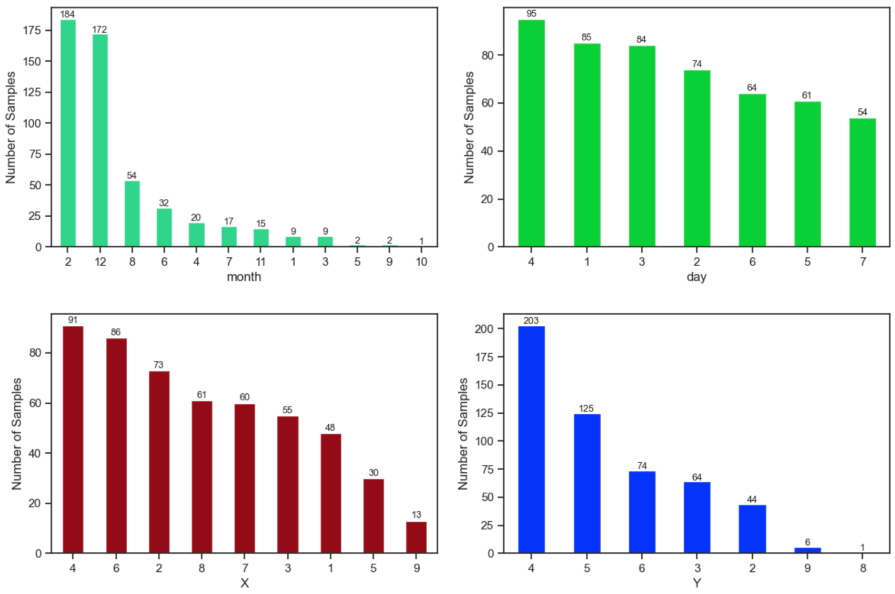

Figure 2. Frequency plots of four features: 'month', 'day' 'X' and 'Y'

2.1.2. Discrete Feature Spare Elements Binning

From Figure 2, we observe that 'month' values 5, 9, and 10 have very few samples, making them sparse elements prone to outliers. To address this, we merge them into a new or existing group with denser elements, e.g., grouping months 5, 9, and 10 into month 11. This reduces overfitting, improves generalization, and lowers one-hot encoding dimensionality and computational cost. Specifically, we group 11, 5, 9, 10 into 'group_mA', 1, 3 into 'group_mB', and in 'Y', we group 2, 8, and 9 into 'group_YA', then apply one-hot encoding.

2.1.3. Feature Selection

For the feature 'rain,' most values are 0, and the few non-zero values each have less than two samples. It can be seen as a combination of a "fixed value," which is useless for regression or classification, and a few outliers, which are potentially harmful to regression or classification.

Thus, this feature is likely to be detrimental overall. Removing it is expected to improve model performance, which is later confirmed by experimental results.

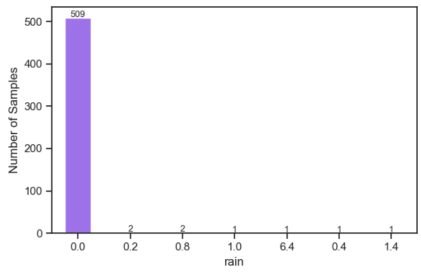

Figure 3. Frequency plots of four features: 'rain'

2.1.4. Features Normalization

Apart from the four discrete features encoded via one-hot, we normalize the remaining features. Since t-SNE and Isomap rely on distances, and PCA uses variance, large-scale features can dominate calculations. Z-score normalization can solve this by ensuring all features are on a similar scale.

2.1.5. Performance Comparison of Features Engineering

The features engineering strategies we discussed above produce benefits in every type of features including original, PCA and Isomap features, shown in Table 1.

Table 1. Performance (RMSE on test dataset) Comparison of Features Engineering Strategies

feature engineering	original features	features from PCA	feaures from Isomap
data0	109.9207	109.9364	109.8263
data1	109.92	109.9386	109.8463
data2	109.9197	109.9316	109.796

data0 = original data + one-hot encoding for the features 'month', 'day', 'X', and 'Y' + Z-score scaling for other features
data1 = data0 + spare element binning
data2 = data1 - the feature 'rain' (remove 'rain')

2.2. Wine dataset & Abalone data set

In Wine and Abalone, we analyze the final feature data from Report 1, review their frequency plots, and ultimately find that there is no need to apply some feature engineering methods (2.1.2 and 2.1.3) used for the forest fire dataset.

3. Representation Learning

We apply t-SNE, PCA, and Isomap to analyze their interpretability (via plots) and prediction impact (via regression models). Since t-SNE is mainly used for visualization (this is also exactly what the assignment requires about t-SNE), we generate a 2D plot for each dataset. To compare linear (PCA) vs. non-linear methods, we select Isomap to assess whether the dataset distribution in R^m space follows a manifold or non-manifold structure to some extent.

3.1. t-SNE

In the forest fire dataset, the 2D t-SNE plots (Figure 4) show a significant effect of normalization. Without normalization (right plot), data points cluster into five dense areas, making them hard to distinguish. With normalization (left plot), the data disperses better, revealing two visible clusters (marked in pink curve) with high burned areas (red and light red dots).

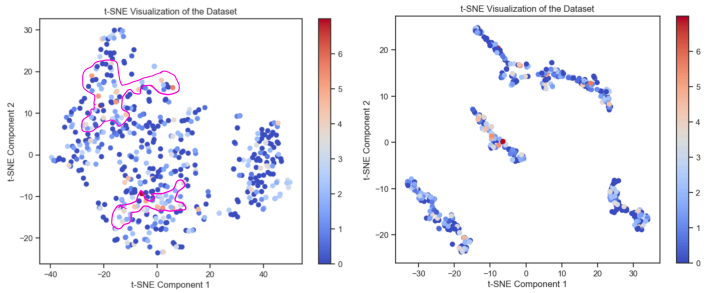

Figure 4. 2D plots by t-SNE on dataset (left) with Z-score normalization (right) without normalization

In Wine, different color points form distinct clusters, making them easy to distinguish, with two separate regions. In Abalone, there are three interesting clusters and red/pink points are concentrated in the upper and lower clusters.

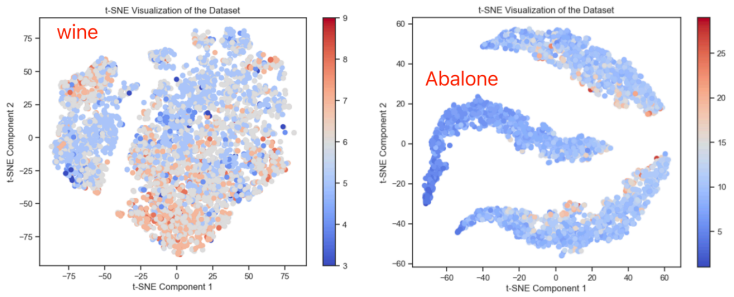

Figure 5. 2D plots by t-SNE on Wine and Abalone datasets

3.2. PCA

The aim of PCA is to reduce the dimensionality of the feature data while obtaining the best approximation for matrix A. By the Eckart-Young theorem[1], the best rank-k approximation (in term of norm) of A is $B = U_k \Sigma_k V_k^T$, where Σ_k retains the top k singular values of A and U_k, V_k contain the first k columns of A' left and right singular matrix U and V, with remaining columns set to zero. The projections of B's column vectors onto each eigenvector of AA^T define a component. If the eigenvector corresponds to the k-th largest eigenvalue, these projections are the k-th principal components of A in PCA. The total number of principal components equals the number of nonzero eigenvalues of AA^T. We have the following properties (our derivation is provided in the Supplemental S1): (i) The variance of i-th principal component equals 1/(n-1) times i-th largest eigenvalue of AA^T. (ii) The total variance of all possible PCA components

remains constant, equaling the total variance of the original data. (iii) The result of using all possible PCA components is $U^T A$ which in geometry, corresponds to only rotating A.

3.2.1. Forest fire dataset

By the PCA properties (i), we know that higher eigenvalues mean higher variance, while lower variance leads to more concentrated data points. The experimental results (Figure 6) align with this, showing that as eigenvalues decrease, data points become more concentrated.

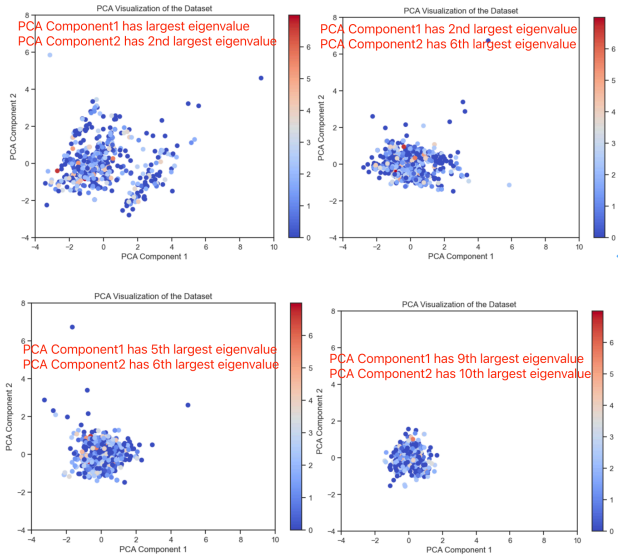

Figure 6. 2D plots by PCA on forest fire dataset with different pair of components

We create a scree plot (Figure 7) to examine cumulative variance, finding that 22 principal components capture over 95% of the variance. Why is the scree plot concave? The question is worth mentioning because it leads us to the analysis on PCA from the perspective of linear algebra. By the properties (ii), the total variance of all components remains constant. When we sort the variances in descending order, the large variances corresponding to the leftmost principal components on the horizontal axis must result in smaller variances for the later components on the right, which makes the plot concave.

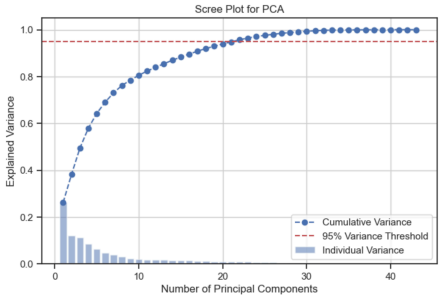

Figure 7. Scree plot of PCA on forest fire dataset

3.2.2. Wine dataset & Abalone dataset

In Wine, reducing to 9 dimensions captures 95% of the variance, while in Abalone, only 4 dimensions are needed. Components 1 and 2 hold the largest variances in both cases. In Wine, data points of the same color cluster together and different colors are well separated, suggesting that PCA effectively differentiates between different targets or feature distributions in the dataset. In Abalone, there are only three values in the component 2 and red points are concentrated on the right of each line. Different colors are also well separated.

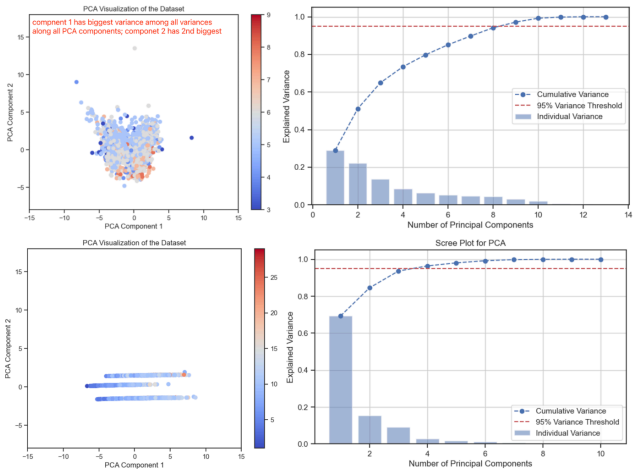

Figure 8. 2D and scree plot of PCA on the Wine (top two) and Abalone (bottom two) datasets.

3.3. ISOMAP

When using Isomap, key questions arise: When is it better to use? If datapoints form a manifold structure (Figure 9), Euclidean distance may cause short circuits (e.g., between x_1 and x_5), while geodesic distance better represents their true relationship. Isomap approximates geodesic distance using piecewise Euclidean distances, making it more effective for manifold-structured data

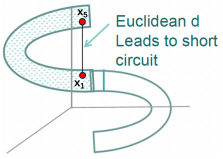

Figure 9. manifold structure

Like PCA, we can create a scree plot for Isomap using cumulative eigenvalues of A^TA. However, does this analogy make mathematical sense? No! In PCA, each principal component has a corresponding variance, and the sum of the variances of all components remains constant. The cumulative variance curve (Scree plot) indicates how many principal components are needed to retain a certain percentage (e.g., 95%) of statistical information (samples' total variance). However, in Isomap, eigenvalues do not represent variance, and the cumulative eigenvalue cannot

represent statistical information of samples, and its percentage has no mathematical meaning. Thus, for Isomap, we conduct experiments in regressors with different dimensions to determine the optimal number of components.

3.3.1. Forest fire dataset

Similarly, we project all data points onto a 2D subspace formed by the first two Isomap components and then plot them in Figure 10. Comparing the 2D plots of Isomap and PCA using the same coordinate scale, we observe that Isomap makes the data points more dispersed than PCA. This suggests that the dataset exhibits more manifold, consistent with our previous discussion.

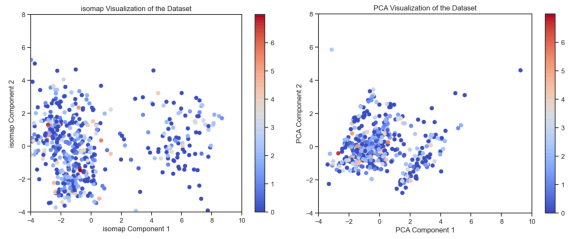

Figure 10. 2D plot of forest fire dataset using (left) Isomap (right) PCA

When the number of principal components exceeds 30, the cumulative variance surpasses 95%. However, considering what we discussed above, in Isomap, we do not use this threshold.

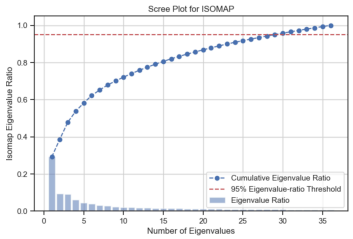

Figure 11. Scree plot of Iosmap on forest fire dataset

3.3.2. Wine dataset & Abalone dataset

The Isomap scree plot of Wine dataset and Abalone data are shown in Figure 12. In Wine, there are two separate regions where data points of the same color cluster together and different colors are well separated, suggesting that Isomap effectively differentiates between different targets or feature distributions in the dataset. In Abalone, it is interesting that the red points are concentrated in the center, while the data points are distributed along three distinct lines with approximately 120-degree angles between them.

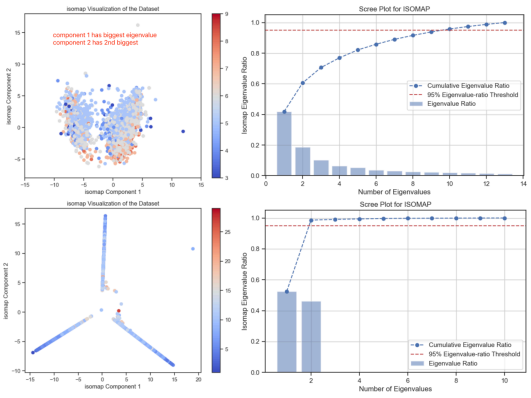

Figure 12. 2D (left) and Scree(right) plot of Wine (top) and Abalone (bottom) dataset using Isomap

4. Split Train & Test Datasets and Perform Cross Validation

First, we divide the data into training and test sets in the ratio 80%:20%. In the training set, we then use 5-fold cross-validation to obtain the best parameter.

In forest fire dataset, we set the hyperparameters in GridSearchCV, k from 1 to 50 by a step of 1 in kNN, 'max_depth': [2, 3, 5, 10], 'n_estimators': [10, 50, 100, 200], 'learning_rate': [0.01, 0.1, 0.2] in Gradient Boosting Regressor (GBR) and 'max_depth': [2, 3, 5, 10, None], 'n_estimators': [10, 50, 100, 200] in Random Forest (RF). We look at the mean and standard deviation of the RMSE, and the standard deviation is acceptable, around 0.1. (Note: We include the Isomap top right plot with cross-validation and different k values of kNN to illustrate the mean and standard deviation, rather than to determine the best setting. The optimal setting is obtained through experiments with different dimensions to find the best number of components.).

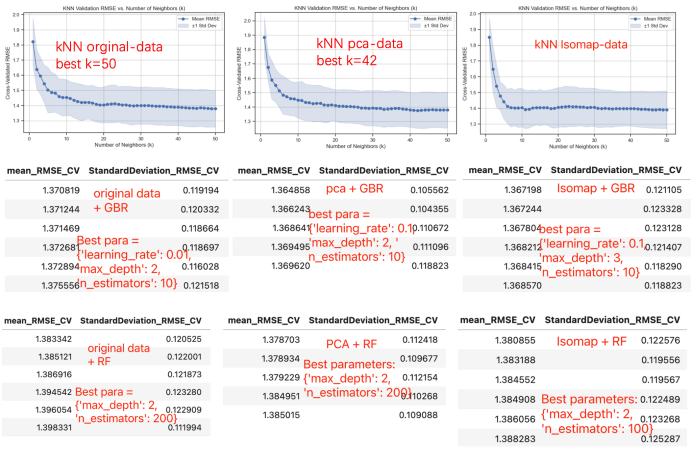

Figure 13. best settings of kNN, GBR and RF with original, PCA and Isomap features from CV.

In Wine dataset and Abalone dataset, we use the same hyperparameters setting in GridSearchCV. To save the space and avoid duplicate description of the report, we give the result directly: all mean and variance are considered, and they are acceptable.

5. Regression Model Performance

5.1. A set of interesting kNN experiments on Forest fire dataset

Before presenting the summary of the best RMSE across different model-dataset combinations, we first analyze a set of interesting experiments that provide a deeper understanding of PCA and Isomap algorithms. We compare the RMSE on the test dataset for a kNN regressor using different numbers of PCA and Isomap components against the original features (red line), as shown in Figure 14.

In PCA plot (left), the last few points almost fall onto the red line. As the number of components approaches the total number of components, the distance between any two data points approaches their original distance in the full-dimensional feature space (our proof is shown in Supplemental S2). As a result, kNN with PCA feature data produces nearly the same predictions as with the original data, leading to a nearly identical RMSE. The final point falls onto the red line because, by property (iii) of PCA in part 3.2, using all PCA components only rotates A without altering distances between data points. Consequently, kNN produces identical predictions, resulting in the same RMSE.

In the Isomap plot (right), when the number of components (NC) > 19, RMSE improves over both original and PCA features (except at NC = 26, 29), suggesting a possible manifold structure in the dataset. Since kNN relies on Euclidean distance, the analysis in part 3.3 shows that Euclidean distance in original and PCA features fails to capture the manifold structure, whereas Isomap features provide a better representation.

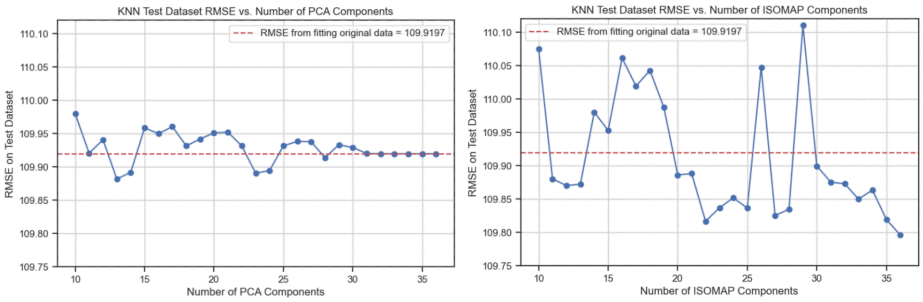

Figure 14. kNN's RMSE on test data with PCA(left), Isomap(right), original features (red line)

5.2. Summary of best RMSE from different models

On test dataset, the best RMSE summary is listed on Table2 (the corresponding settings are provided in Supplemental S3), the best RMSE in each dataset is marked in red. There are some interesting points on the table:

(1) In kNN, across all datasets, Isomap features achieve the lowest RMSE, outperforming both original and PCA features (Rows 1, 4, 7). However, in tree-based models (Random Forest and Gradient Boosting), Isomap features do not consistently perform better. This is because kNN relies on FEATURE distances between datapoints, which Isomap improves by using piece-wise Euclidean distances in manifold structure, while tree-based models split based on MSE reduction or residuals on TARGET rather than distances between datapoints, so better distance representation from Isomap do not necessarily improve RF and GB performance. As a result, Random Forest and Gradient Boosting benefit from Isomap in the forest fire dataset (row 2,3) but performs worse in Isomap in other datasets (row 5,6,8,9). This also shows that three datasets are likely to have manifold structure.

(2) Original feature data achieves lower RMSE than PCA feature data (95% threshold). We know that by PCA property (iii) in part 3.2, a 100% threshold means the original feature matrix is only rotated in space, leading to identical predictions across models. A 95% threshold, however, implies some information loss, making regressors perform worse than on the original data. However, PCA does not only discard information—it may also introduce complex, less interpretable components. The combined effect of these factors can either improve (Figure 14 left, when the number of PCA components = 13, 14, 23, 24, 28) or worsen regressors' performance (Table 2). Additionally, tree-based models generally perform better on original data, as splits rely on interpretable features.

(3) The performance difference between GB and RF varies across datasets. In the forest fire dataset, when using PCA or Isomap features with kNN, GB performs better (row 2, 3), while in other cases, RF achieves better results. This can be attributed to the fact that RF, by averaging multiple trees, reduces variance and provides more stable predictions. On the other hand, GB focuses on reducing residuals iteratively, which can lead to better performance in some cases but also higher sensitivity to noise.

(4) It takes more time to fit GB than RF because GB builds trees sequentially, with each tree correcting the residuals of the previous one in a step-by-step manner. In contrast, RF trains multiple trees independently and in parallel, making it more time efficient.

Table 2. Best RMSE (test dataset) on different models with different features

	regressor	original features	features from PCA (95% cumulative variance)	features from Isomap	row index
forest fires	knn	109.9197	109.9316	109.796	1
	RandomForest (RF)	109.9134	110.0096	109.9051	2
	GradientBoosting (GB)	109.9847	109.9934	109.8947	3
wine	knn	0.7671	0.7845	0.6786	4
	RandomForest (RF)	0.5609	0.5822	0.5982	5
	GradientBoosting (GB)	0.5854	0.6021	0.6026	6
abalone	knn	2.6801	2.8442	2.3034	7
	RandomForest (RF)	2.2261	2.448	2.2806	8
	GradientBoosting (GB)	2.2766	2.4535	2.2809	9

References:

[1]. https://www.youtube.com/watch?v=Y4f7K9XF04k&list=PLUl4u3cNGP63oMNUHXqIUcrkS2PivhN3k&index=11

Supplemental

S1. the derivation of properties of PCA.

let $U=[u_1,\cdots u_m]$ be left signular matrix of A

Separate each column a_j of A into all components along $u_1,\cdots,u_m$

$$\sum_1^n \|a_j\|^2 = \sum_1^n |a_j^T u_1|^2 + \cdots + \sum_1^n |a_j^T u_i|^2 + \cdots + \sum_1^n |a_n^T u_m|^2$$

$$\because \ |a_j^T u_1|^2 = (a_j^T u_1)^T (a_j^T u_1) = u_1^T (a_j a_j^T) u_1$$

$$\therefore \sum_1^n |a_j^T u_1|^2 = u_1^T \sum_1^n (a_j a_j^T) u_1 = u_1^T (AA^T) u_1$$

$$= u_1^T (U \Sigma^2 U^T) u_1 \quad (\because \text{SVD of } A)$$

$$= (u_1^T U)\, \Sigma^2 (U^T u_1) = [1,0\cdots 0]\, \Sigma^2 \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix} \quad (\because \text{property of orthogonal matrix})$$

$$= \lambda_1 \quad (\text{the 1st entry on diagonal}(\Sigma^2))$$

where λ_1 is the 1st largest eigenvalue of AA^T

$$\therefore \sum_1^n \|a_j\|^2 = \lambda_1 + \cdots + \lambda_i + \cdots + \lambda_k = \sum_{i=1}^k \lambda_i \quad (k \le \min(m,n))$$

where λ_i is the ith largest eigenvalue of AA^T

By definition of variance, we know:

$$\frac{\lambda_i}{n-1} = \frac{\sum_{j=1}^n |a_j^T u_i|^2}{n-1} = \text{variance of samples projection on } u_i$$

$$= \text{Squared norm of } i\text{th largest component}$$

We know that sum of variances of all (original) samples is $\frac{1}{n-1}\sum_1^n \|a_j\|^2$

then $\frac{1}{n-1}\sum_1^n \|a_j\|^2 = \sum_1^n \frac{\lambda_i}{n-1}$

S2. Proof that as the number of components approaches n, the distance between any two data points approaches their original distance in the full-dimensional feature space.

$$[x_1 \cdots x_n] = A \overset{SVD}{=} U\Sigma V^T = \sigma_1 u_1 v_1^T + \cdots + \sigma_r u_r v_r^T$$

x_i, u_i, v_i is column vector of A, U, V

σ_i is the eigenvalue of AA^T, also diagonal entry of Σ^2

$\sigma_1 \to \sigma_r$ are sorted in descending order.

If we drop the components with smallest eigenvalues, and the distance between two datapoints $x_i^{(p)}, x_j^{(p)}$ is d_{ij}

then:

$$d_{ij}^2 = \left\| X_i^{(p)} - X_j^{(p)} \right\|_2^2$$

$$= \left\| \left(x_i - (\sigma_t C_t^{(i)} u_t + \cdots + \sigma_r C_r^{(i)} u_r) \right) - \left(x_j - (\sigma_t C_t^{(j)} u_t + \cdots + \sigma_r C_r^{(i)} u_r) \right) \right\|_2^2$$

where C_t's are constant values.

When $\sigma_t, \cdots, \sigma_r$ are very small, we have:

$d_{ij}^2 \approx \| x_i - x_j \|_2^2$ = the distance of x_i, x_j

x_i, x_j are the original datapoints of $x_i^{(p)}, x_j^{(p)}$ before PCA

S3. Summary of models' settings with best RMSE

dataset	regressor	original features	features from PCA (95% cumulative variance)	features from Isomap	row index
forest fires (dim = 36, after feature engineering)	knn	k=50	k = 43	dim=36,k=47	1
	RandomForest (RF)	{'max_depth': 2, 'n_estimators': 200}	{'max_depth': 2, 'n_estimators': 200}	dim = 33, {'max_depth': 2, 'n_estimators': 100}	2
	GradientBoosting (GB)	{'learning_rate': 0.01, 'max_depth': 2, 'n_estimators': 10}	{'learning_rate': 0.1, 'max_depth': 2, 'n_estimators': 10}	dim = 32, {'max_depth': 3, 'n_estimators': 10}	3
wine (dim = 12, after feature engineering)	knn	k = 15	k = 8	dim = 8, k = 31	4
	RandomForest (RF)	{'max_depth': None, 'n_estimators': 200}	{'max_depth': None, 'n_estimators': 200}	dim = 7 {'max_depth': None, 'n_estimators': 200}	5
	GradientBoosting (GB)	{'learning_rate': 0.1, 'max_depth': 10, 'n_estimators': 100}	{'learning_rate': 0.1, 'max_depth': 10, 'n_estimators': 100}	dim = 8 {'learning_rate': 0.1, 'max_depth': 10, 'n_estimators': 200}	6
abalone (dim = 10, after feature engineering)	knn	k = 11	k = 32	dim = 9, k = 20	7
	RandomForest (RF)	{'max_depth': 10, 'n_estimators': 200}	{'max_depth': 10, 'n_estimators': 200}	dim = 10 {'max_depth': 10, 'n_estimators': 200}	8
	GradientBoosting (GB)	{'learning_rate': 0.2, 'max_depth': 3, 'n_estimators': 50}	{'learning_rate': 0.1, 'max_depth': 2, 'n_estimators': 100}	dim = 9 {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 50}	9

The derivation of MLE and MAP:

MLE estimator

$$L(b \mid x_1, \cdots x_i \cdots x_N) = \prod_{i=1}^{N} f(x_i) = (ab)^N \left(\prod_{i=1}^{N} x_i\right)^{a-1} \left(\prod_{i=1}^{N} (1-x_i^a)\right)^{(b-1)}$$

$$\log L(b) = N \log(ab) + (a-1)\sum_{i=1}^{N} \log x_i + (b-1)\sum_{i=1}^{N} \log(1-x_i^a)$$

$$\text{let } \frac{d(\log L(b))}{d(b)} = 0 \quad \Rightarrow \frac{N}{b} + \sum_{i=1}^{N} \ln(1-x_i^a) = 0$$

$$\{\log(\cdot) = \ln(\cdot)\}$$

$$\Rightarrow b = \frac{N}{-\sum_{i=1}^{N} \ln(1-x_i^a)}$$

MAP estimator for b.

$$b_{map} = \arg\max_b P(b|D) = \arg\max_b \left(\frac{P(D|b) \cdot P(b)}{P(D)}\right)$$

$$(D \text{ is } N \text{ samples}, x_1, \cdots, x_i, \cdots x_N)$$

$$\Leftrightarrow \arg\max_b \{\log P(D|b) + \log P(b) - \log P(D)\}$$

$$\Leftrightarrow \arg\max_b \left\{\log \prod_{i=1}^{N} f(x_i|b) + \log P(b)\right\} \quad (\because a \text{ is known and } P(D) \text{ is constant (value)})$$

$$= \arg\max_b \left\{\log L(b) + \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(b-w)^2}{2\sigma^2}\right)\right)\right\}$$

$$= \arg\max_b \left\{N \log b + (b-1)\sum_{i=1}^{N} \log(1-x_i^a) - \frac{(b-w)^2}{2\sigma^2} + C\right\}$$

$$= \arg\max_b F(b)$$

method 1: $\text{let } \frac{dF(b)}{db} = \frac{N}{b} + \sum_{i=1}^{N} \log(1-x_i^a) - \frac{b-u}{\sigma^2} = 0$

$$\Leftrightarrow N + b \cdot \sum_{i=1}^{N} \log(1-x_i^a) - \frac{(b-u)b}{\sigma^2} = 0$$

$$\Leftrightarrow N\sigma^2 + b \cdot \sigma^2 \sum_{i=1}^{N} \log(1-x_i^a) - b^2 + ub = 0$$

$$\Leftrightarrow b^2 - b \cdot \left(u + \sigma^2 \sum_{i=1}^{N} \log(1-x_i^a)\right) - N\sigma^2 = 0$$

$$b = \frac{u + \sigma^2 \sum_{i=1}^{N} \log(1-x_i^a) \pm \sqrt{\left(u + \sigma^2 \sum_{i=1}^{N} \log(1-x_i^a)\right)^2 + 4N\sigma^2}}{2}$$

method 2 (gradient descent): $b^{(t)} = b^{(t-1)} - \frac{dF(b^{(t-1)})}{db^{(t-1)}} \cdot \gamma$ (γ is learning rate)

iterative it until $|b^{(t)} - b^{(t-1)}|$ or $|F(b^{(t)}) - F(b^{(t-1)})|$

or $\left|\frac{dF(b^{(t)})}{db^{(t)}} - \frac{dF(b^{(t-1)})}{db^{(t-1)}}\right|$ arrives at a value $\leq$ threshold we set, then we get final b.