Milestone 2 Report

Team members: Ninglin Huang, Yu Wang, Qiuyi Zhang

Data Preprocessing

1. Find any errors or missing values in our data

(1) Data type errors

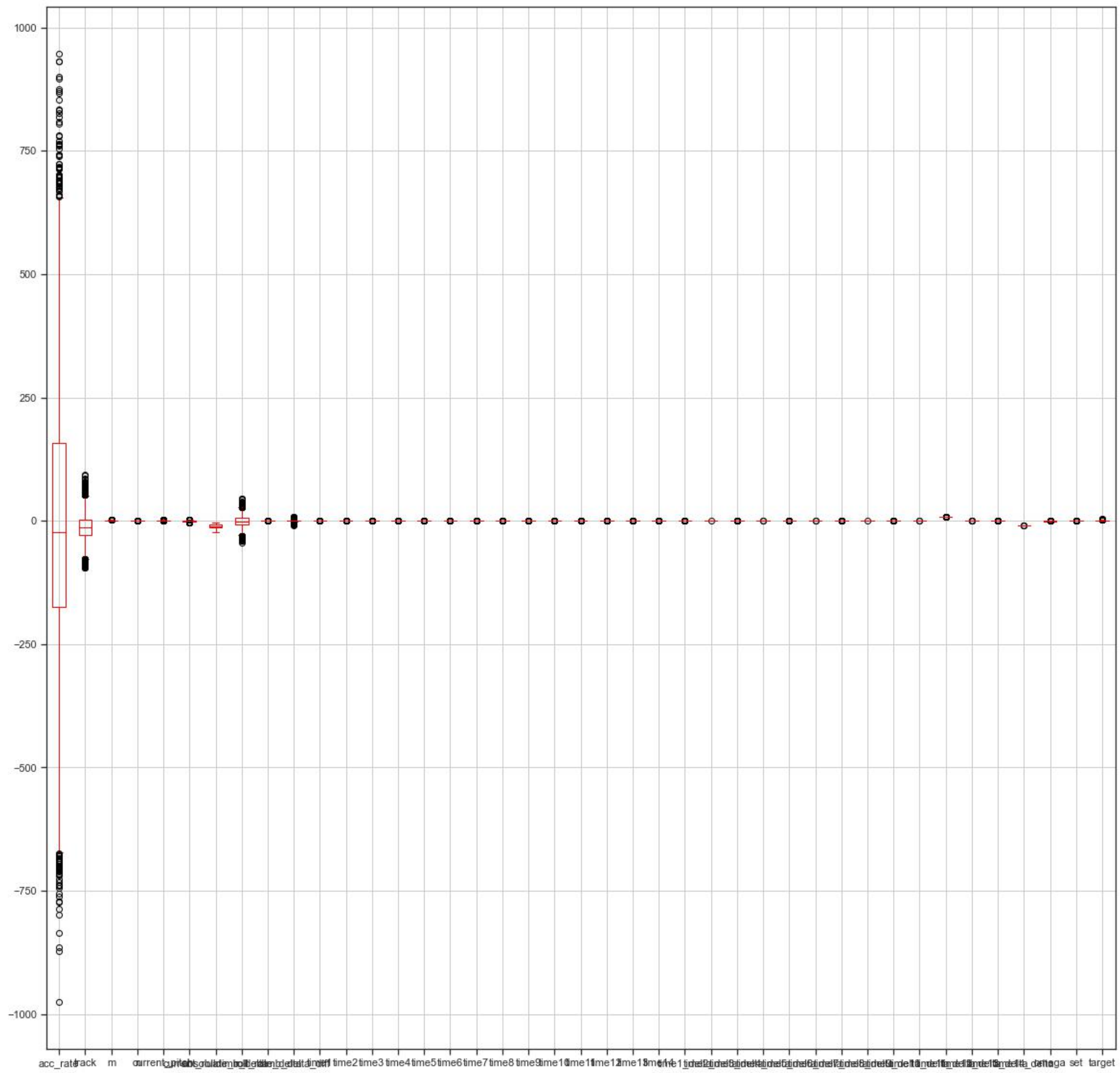
```
df_train.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8250 entries, 0 to 8249
Data columns (total 41 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   acc_rate          8250 non-null   int64
 1   track             8250 non-null   int64
 2   m                 8250 non-null   float64
 3   n                 8250 non-null   float64
 4   current_pitch     8250 non-null   float64
 5   current_roll      8250 non-null   float64
 6   absoluate_roll    8250 non-null   int64
 7   climb_delta       8250 non-null   int64
 8   roll_rate_delta   8250 non-null   float64
 9   climb_delta_diff  8250 non-null   float64
 10  time1             8250 non-null   float64
 11  time2             8250 non-null   float64
 12  time3             8250 non-null   float64
 13  time4             8250 non-null   float64
 14  time5             8250 non-null   float64
 15  time6             8250 non-null   float64
 16  time7             8250 non-null   float64
 17  time8             8250 non-null   float64
 18  time9             8250 non-null   float64
 19  time10            8250 non-null   float64
 20  time11            8250 non-null   float64
 21  time12            8250 non-null   float64
 22  time13            8250 non-null   float64
 23  time14            8250 non-null   float64
```

```
 24  time1_delta       8250 non-null   float64
 25  time2_delta       8250 non-null   float64
 26  time3_delta       8250 non-null   float64
 27  time4_delta       8250 non-null   float64
 28  time5_delta       8250 non-null   float64
 29  time6_delta       8250 non-null   float64
 30  time7_delta       8250 non-null   float64
 31  time8_delta       8250 non-null   float64
 32  time9_delta       8250 non-null   float64
 33  time10_delta      8250 non-null   float64
 34  time11_delta      8250 non-null   float64
 35  time12_delta      8250 non-null   float64
 36  time13_delta      8250 non-null   float64
 37  time14_delta      8250 non-null   float64
 38  omega             8250 non-null   float64
 39  set               8250 non-null   float64
 40  target            8250 non-null   float64
dtypes: float64(37), int64(4)
memory usage: 2.6 MB
```

There is no data type errors.

(2) Outliers

a.Draw boxplots of all features:



We can find three columns have obvious outliers, they are acc_rate, track and climb_delta.

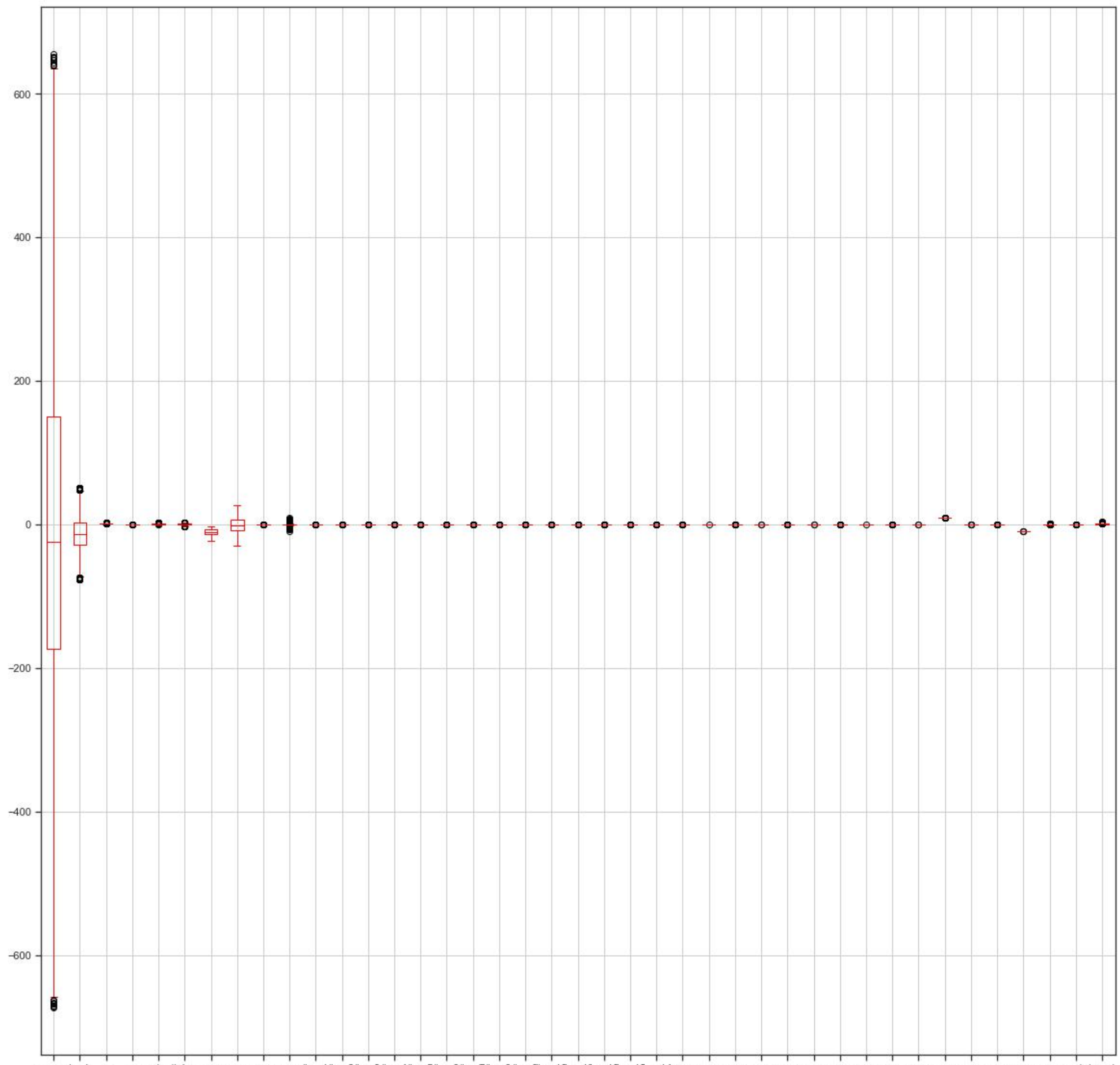
b. Drop outliers:

acc_rate:

```
# acc_rate
boxplot = df_train.boxplot(column=['acc_rate'])
whiskers = boxplot.get_lines()[1:3]
lower_whisker = whiskers[0].get_ydata()[1]
upper_whisker = whiskers[1].get_ydata()[1]
outliers = df_train[(df_train['acc_rate'] < lower_whisker) | (df_train['acc_rate'] > upper_whis
#print(outliers)
df_train_clean = df_train[(df_train['acc_rate'] >= lower_whisker) & (df_train['acc_rate'] <= up
```

And perform the same steps for track and climb_delta.

c. After dropping the outliers:



Now the remaining data is 7926 rows.

(3) Duplicated rows

```
# duplicated rows
duplicated_rows = df_train_clean.duplicated()
print(duplicated_rows)

df_train_clean = df_train_clean.drop_duplicates()
```

```
0       False
1       False
2       False
3       False
4       False
        ...
8244    False
8246    False
8247    False
8248    False
8249    False
Length: 7926, dtype: bool
```

There is no duplicated rows.

(4) Missing values

```
# missing values
missing_values = df_train_clean.isnull().sum()
print(missing_values)
```

```
acc_rate            0
track               0
m                   0
n                   0
current_pitch       0
current_roll        0
absoluate_roll      0
```

There is no missing values.

2. Consider models that learn with gradient descent–based methods. Why would we want to rescale our data in this case?

The gradient descent will converge more quickly to the minima if the features are on a similar scale. When features are all on the same scale, it's easy to compare how much they contribute to the model.

3.Consider models that perform regularization according to the norm of the parameters. Why would we want to rescale our data in this case?
Regularization penalizes the model for large coefficients. If the scales of the input features are different, the regularization term will have a bigger effect on the features with larger scales, leading to biased estimates. Rescaling data makes sure that all features have the same scale, so the regularization term can equally treat each feature.

4.Consider models that learn according to distance measures (SVM, k-nearest neighbors, etc.). Why would we want to rescale our data in this case?
If features have different scales, those with larger magnitudes may dominate the distance calculations. Rescaling data makes sure that all features impact the distance calculation equally, which makes the model can better capture the patterns in the data. It can also make the distance computation faster in some cases.

5. Choose GaussianProcessRegressor model and perform normalization on the data, train the model on normalized data with k cross-validation and compare the performances (both in-sample and out-sample) between before and after normalization using a paired t-test:

```
Paired t-test results for In-sample:
t-statistic: -36.026334, p-value : 4.841860e-11
Paired t-test results for Out-sample:
t-statistic: -6.116298, p-value : 1.757377e-04
```

In both cases, the p-values are much smaller than 0.05, so we can reject the null hypothesis. Thus there is a significant difference in the performance of the GPR model before and after normalization.

In both cases, the negative t–statistics show that the MSE after normalization is smaller than the MSE before normalization. This suggests that performance is now better after normalization.

This finding makes sense because the Gaussian Process Regressor relies on distance measures, and normalization helps deal with the problem that different feature scales can affect the performance of the model.

Feature Selection

1.First, briefly explain an advantage and a disadvantage of our data having many features. Why might we want to use only a subset of the features?

Advantage: Having many features means rich sources of information and the potential to learn more complex patterns from the data;
Disadvantage: Too many features will increase the training time of the model and make the computation more complicated;

Using a subset of features increases the interpretability of the model, and the model will focus on features that are more relevant to the prediction target; it can also avoid overfitting; reduce training costs and time; improve the accuracy of the model.

2.Recall the statistics and KDE of each feature from Milestone 1. Based on these results, what are some features (at least 3) you might want to discard? Explain your decision for each feature you wish to discard. If there aren't any, explain why.
The features that I want to discard are:
time2_delta
time4_delta
time6_delta

time8_delta
time14_delta
The standard deviations and ranges of these features are too small to provide much information, and there are no significant peaks in the KDE plots.

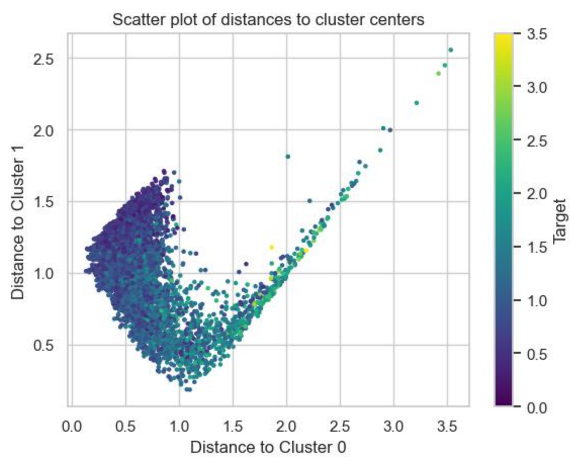
3.Recall the correlation between each feature and the target from Milestone 1. Based on these results, what are some features you might want to keep (at least 1) and some you might want to discard (at least 1)? Explain your decision for each feature you wish to either discard or keep.
keep: absoluate_roll, time1, time5
These features have high correlations with target.
discard: time10_delta, time12_delta, time2_delta
These features have loow correlations with target.

4.Recall the correlation between each of the features from Milestone 1. Based on these results, find pairs of features (at least 1) from which you might want to only select one. Explain your decision for the pair you have chosen. Choose one of the features to keep and one of the features to discard, and explain your decision.

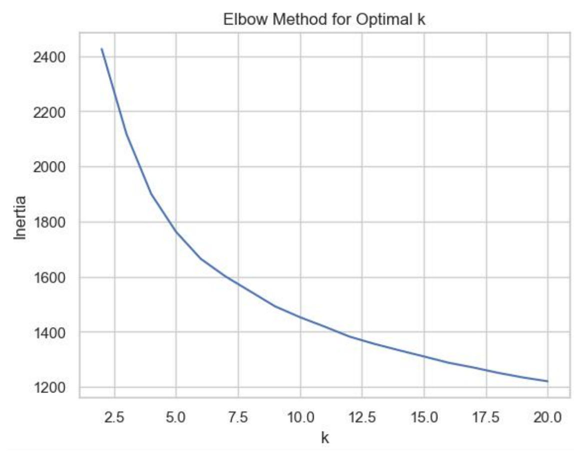
Paired of features: time1, time2
These two features are highly correlated and may provide redundant information.
Keep time1 and discard time2:
Compare to time2, time1 is more correlated with target, so it is more important.
Thus we can keep time1 and discard time2

Clustering

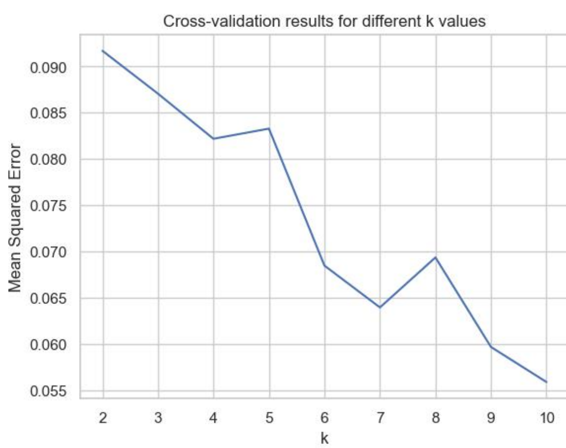
First, we use the clustering method to divide the data into two distinct groups. The distance between each standardized data point and the two cluster centers is shown in this image, with color indicating the axis of the corresponding y value. Analyzing this image, we can see the obvious trend of the data, which is that when the distance from the point to the center of cluster 0 is greater than the distance from the point to the center of cluster 1, the y value of the data point corresponds to greater than 1.5, and the y value of the data point is larger when it is further away from the two cluster centers. On the contrary, the y value is considerably lower than 2.0. This demonstrates that the information obtained through clustering has a connection with the y value.

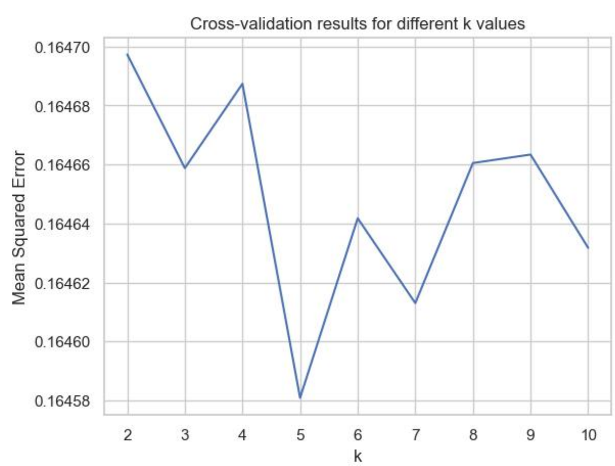


We first used Elbow's method to determine the number of clusters k, attempting to find the value of k that minimizes the sum of squares gain within the cluster. However, no obvious point of turning was found.



After clustering the data, we perform linear regression using the distance of each data point from the cluster center point as a variable and the target as the y value, and cross-validate each k value to obtain the MSE of linear regression. The graph with normalized data is shown below:

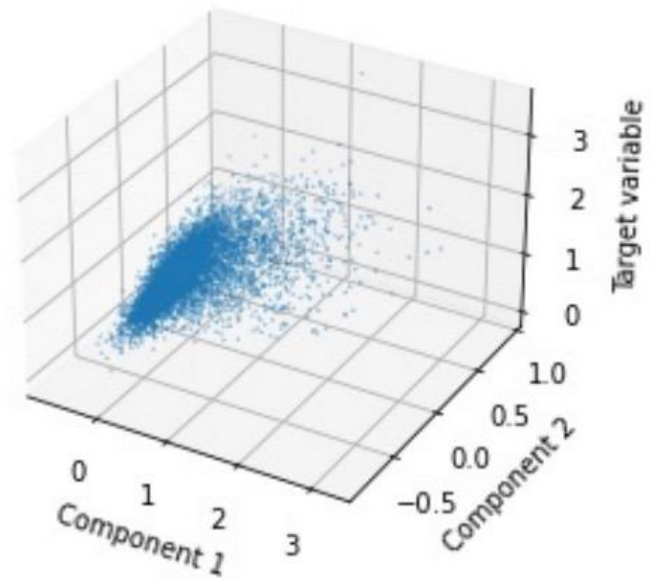




MSE is not stable as k increases, but the trend is still downward. Then, with unstandardized data, we tried again, and the MSE dropped significantly when k=5, so we chose k=5.

PCA

1.Perform PCA, keeping two components. Then, plot a 3d scatter plot where the x, y axes correspond to the two components and the z axis corresponds to the target variable. Interpret this plot.

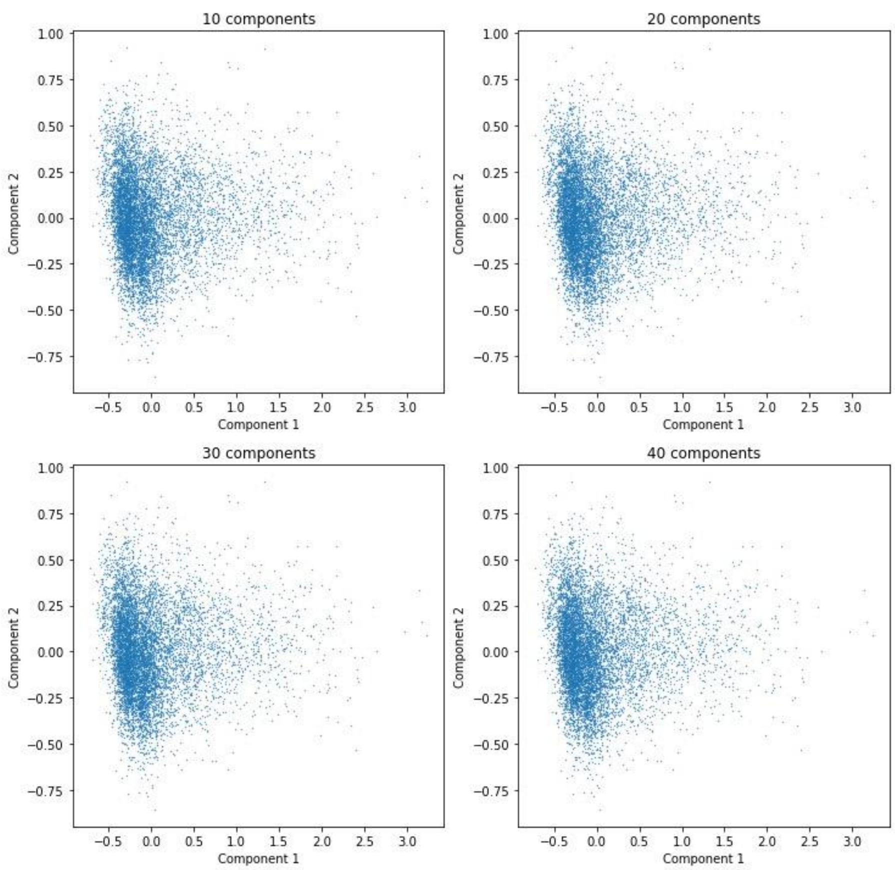


After performing PCA with only two components, the 3d scatter plot could be drawn as:

In this scatterplot, the x and y coordinates of each point correspond to the two principal components in the dataset, while the z coordinate represents the target variable. Therefore, we can use this graph to see the relationship between the principal components and the target variable. For example, if we observe that the z coordinates are concentrated in an area where the x and y coordinates are small, then this is an indication that the principal components in the dataset may not explain variation in the target variable well, or that there is a strong correlation between the principal components . Conversely, if we see scatter across the graph space, then this could mean that the principal components explain variation in the target variable well, or that there is a weak correlation between the principal components.

Observing the plot above, we could regard the scatter plot are locate in the whole space, which meansthat the principle components here could explain out target variable well, and the two principle components have low correlation between each other.

2.Perform PCA for the following number of components: 10, 20, 30, 40.



Performing the PCA for different numerical components, and drawing the relationship of the first component and second component, we could find that the relationship of the first and second components we choose do not change so much even though the the number of components change a lot.

3. Choose one model that you used from Milestone 1. For each PCA-transformed version of the dataset, perform k cross-validation on the data with the model. Compare the performances between each version and the original standardized dataset.

The model we choose here is the K Nearest Neighbor model, which is a continue analysis based on the above part. In the first step, let's us see the result based on the two component.

score	Unnormalized data	Normalized data	Data after PCA
Train	0.08	0.02	0.02
Test	0.08	0.04	0.04

Given by the table, we could find that the normalization process improve the model a lot since the score decrease. And when we use the data after PCA, the result improve when comparing to the unnormalized data, and the score given by the data after PCA is approximately the same with the score given by the normalized data. Because the score here are similar to each other, we could derive the conclusion that only two components in PCA could also get a perfect result. And only two components here could give us a very simple explanation.

In the second step, we also consider the case that the number of components is equal to 10,20,30,40 under the knn model.

Train score	2	10	20	30	40
Normalized	0.02	0.02	0.02	0.02	0.02

PCA	0.02	0.02	0.02	0.02	0.02

Test score	2	10	20	30	40
Normalized	0.04	0.04	0.04	0.04	0.04
PCA	0.04	0.04	0.04	0.04	0.04

P value	2	10	20	30	40
In-sample	3.500e-17	0.014	0.196	0.181	nan
Out-sample	1.86e-09	0.929	0.367	0.408	nan

Given by the table above, we could derive the conclusion as following:

1. For the case which have larger than 10 principal components, using PCA dimensionality reduction do not significantly improve the performance of the model on the training set.
2. When we used 2 principal components, we found that the training and test scores of the model were good and statistically significant after using PCA. This shows that PCA can help us reduce the number of input features while retaining most of the information in the data, thereby improving the performance of the model. Furthermore, we noticed that applying PCA before normalizing the data leads to better performance.
3. When we used 10, 20, 30, and 40 principal components, we did not observe significant performance improvement, suggesting that on this dataset, using 2 principal components is sufficient to capture most of the information in the data. But since the p value for component is equal to 2 is very small, in this case we

should reject the null hypothesis that pca or not do not influence the result. That is, when the component is 2, the pca will influence the result.

4. All the p value for in sample and out sample t test are small. Furthermore, we also found that using normalized data performed better than unnormalized data in these cases. It should be noted that when using 40 principal components, we cannot perform a t-test.

5. Since when the component is 2, the score is nice but the p is small, testing more values of component which have a smaller value would be a better choice comparing to the 10,20,30,40.