IML assignment 1

Nabiullin Damir

d.nabiullin@innopolis.university

▼ Preprocessing

▼ Prepare $var7$

I decided to handle column $var7$. Therefore, I created my own function `date_to_seconds`. I made it to don't use implemented in python datetime, which can raise exception on non valid date. While my function will convert it correctly to seconds. This function returns time in seconds from 1970-01-01. As we don't know whether the data depends on the position, I decided that this time is computed in UTC.

▼ Encoding (Answer to the question 2.1.2)

As we discussed during labs: For categorical variables where ordinal relationship exists, we use Ordinal encoding. But in our task in column $var3$ we have countries, therefore One-Hot encoding looks more appropriate. However, in our task we will have a problem when using this type of encoding.

Before encoding, I checked how many different countries we have in column $var3$. I found that in this task we have 236 unique countries in column $var3$.

If we will use One-Hot encoding, then new data frame will have more than 200 features (excluding column 7 encoding). On this amount of features it is hard/impossible to use polynomial regression with degree more than 2.

Moreover, I tried both Ordinal and One-Hot encoders. For Ordinal encoding linear regression had error less than for One-Hot regression.

Therefore, I used **Ordinal encoding**.

▼ Data imputation (Answer to the question 2.1.1)

To find the missing values of the $var4$ column I tried the Linear and Polynomial regression models.

Firstly, I decided that target column will increase our prediction of $var4$. Therefore, I divided data into 2 parts (where $var4$ is NAN and where $var4$ is not NAN) including $target$ column. Moreover, I divided each data partition into target and features, where target is $var4$ column and features all other columns.

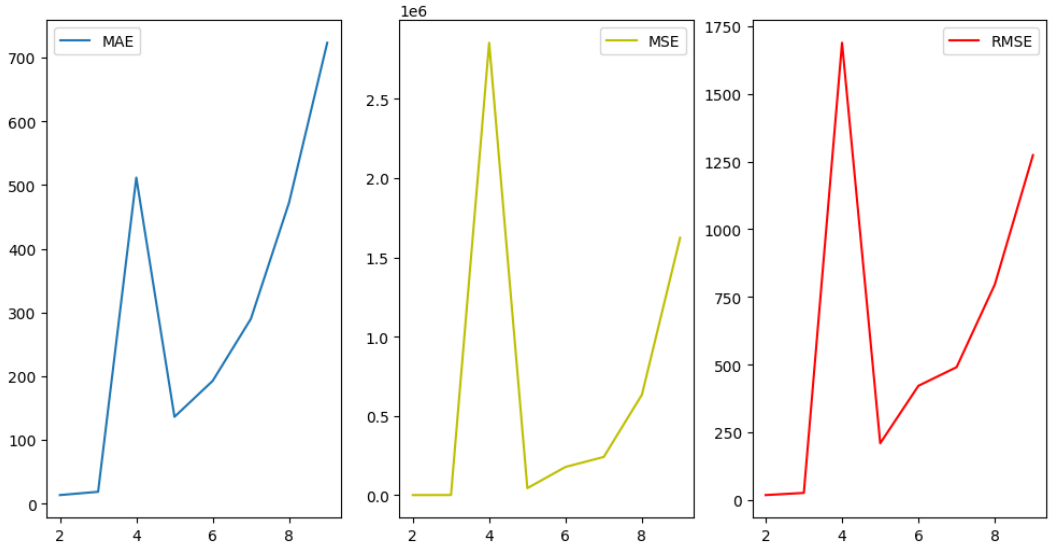
▼ Linear regression

When i tried Linear regression model, I got:

Mean Absolute Error: 16.76
Mean Squared Error: 462.42
Root Mean Squared Error: 21.50

▼ Polynomial regression

Moreover, I tried Polynomial regression model with degrees in range from 2 to 9. Errors are represented on image:



I got the smallest error with degree equal to 2. For this degree I have such errors:
Mean Absolute Error: 13.54
Mean Squared Error: 321.4
Root Mean Squared Error: 17.93

According to errors, polynomial regression with degree 2 is the most optimal model to fill missing values.

Moreover, according to errors:

The Linear regression suffers from underfitting.
The Polynomial model with degree more than 2 suffers from overfitting.

The **Polynomial model with degree 2** describes the function closest to the test values. Therefore, I used it.

▼ Implementing PCA

Before using PCA, I transformed data using StandardScaler to have zero mean value.

Moreover, I implemented my own PCA class.

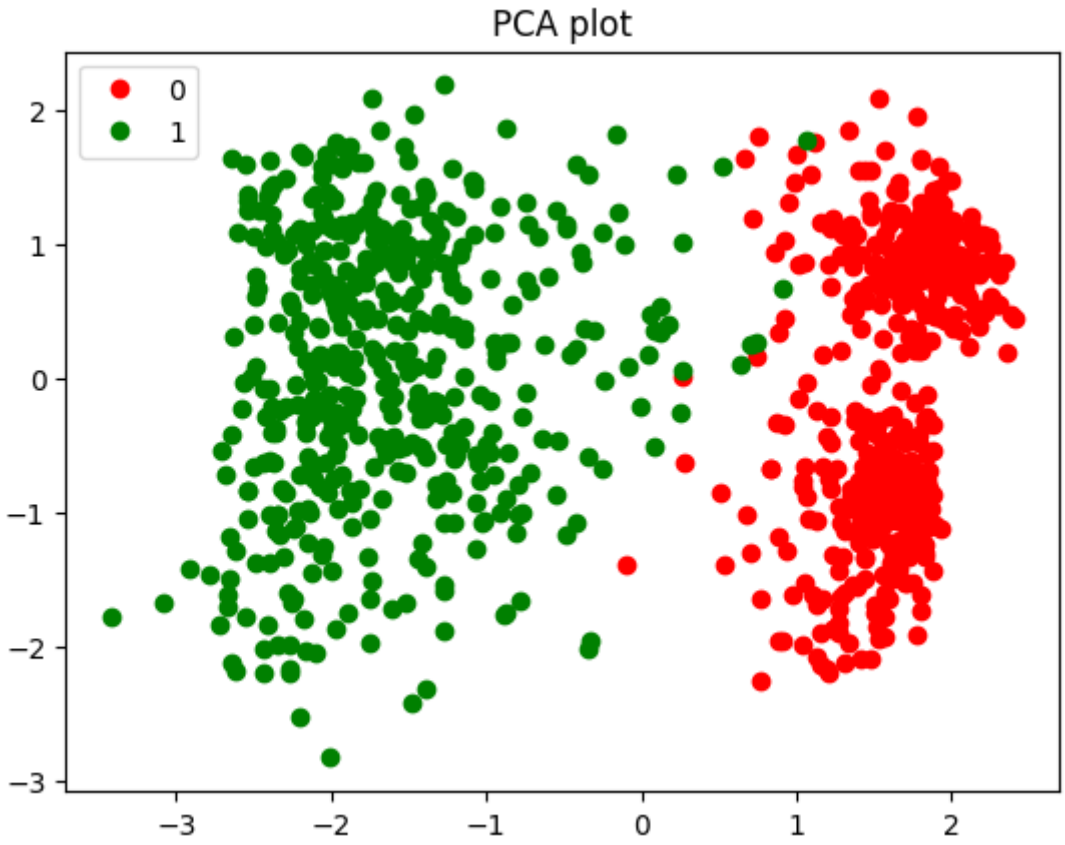
In this class I implemented constructor, that takes expected dimension or percent of variance. My PCA understand by itself what is given as parameter in constructor by checking type.

Function `fit` calculates eigenvectors and explained variance for future use. It sorts eigenvectors by eigenvalues and takes needed number of the best eigenvectors. Moreover, this function can plot eigenvalues and print eigenvectors if needed (put corresponding parameters).

Function `transform` computes data in new dimension by multiplying given data with eigenvectors.

Function `explained_variance_ratio` returns explained variance.

When data is scaled and PCA is implemented, I decided to plot the computed data.



▼ Training

Before implementing ML algorithms I divided preprocessed data on target and features. Secondly, I divided it on training and testing set. Finally, I scaled it to reduce scatter and speed up learning.

▼ Without PCA

▼ Logistic Regression

Firstly, I checked Logistic Regression on KFold Cross-Validation with k equal to 3 with training set. I got Cross-Validation score of accuracy equal to `0.95812`.

Secondly, I checked this model on testing set and got score of accuracy equal to `0.98378`.

▼ KNN

Firstly, I checked KNN model on KFold Cross-Validation with k equal to 3 with training set. I got Cross-Validation score of accuracy equal to `0.95946`.

Moreover, I found the best K for my model on range from 1 to 10. K is equal to 9.

Secondly, I checked this model with K equal to 9 on testing set and got score of accuracy equal to `0.97838`.

▼ Naive Bayes

Firstly, I checked Naive Bayes model on KFold Cross-Validation with k equal to 3 with training set. I got Cross-Validation score of accuracy equal to `0.95273`.

Secondly, I checked this model with K equal to 9 on testing set and got score of accuracy equal to `0.98378`.

According to Cross-Validation without PCA - the best model is KNN with k equal to 9. But other models are so close to KNN.

But on testing set Logistic Regression and Naive Bayes were a little bit better than KNN.

▼ With PCA

Before using PCA, I created object of PCA with expected percent of variance equal to `0.8`**.** In my case, such PCA took 5 the best eigenvectors.

▼ Logistic Regression

After checking model on Cross-Validation, I got score of accuracy equal to `0.95407`.

Moreover, I got score of accuracy on testing set equal to `0.96757`.

▼ KNN

After checking model on Cross-Validation, I got score of accuracy equal to `0.95137`. In addition, I found the best K for my model on range from 1 to 10. K is equal to 5.

However, I got score of accuracy on testing set equal to `0.98919`.

▼ Naive Bayes

After checking model on Cross-Validation, I got score of accuracy equal to `0.95273`.

Moreover, I got score of accuracy on testing set equal to `0.98378`.

According to Cross-Validation with PCA - the best model is Logistic Regression. But other models are so close to this model.

But on testing set KNN showed the best result among all tests.

▼ The Best Model (Answers to the questions 2.2.1 and 2.2.4)

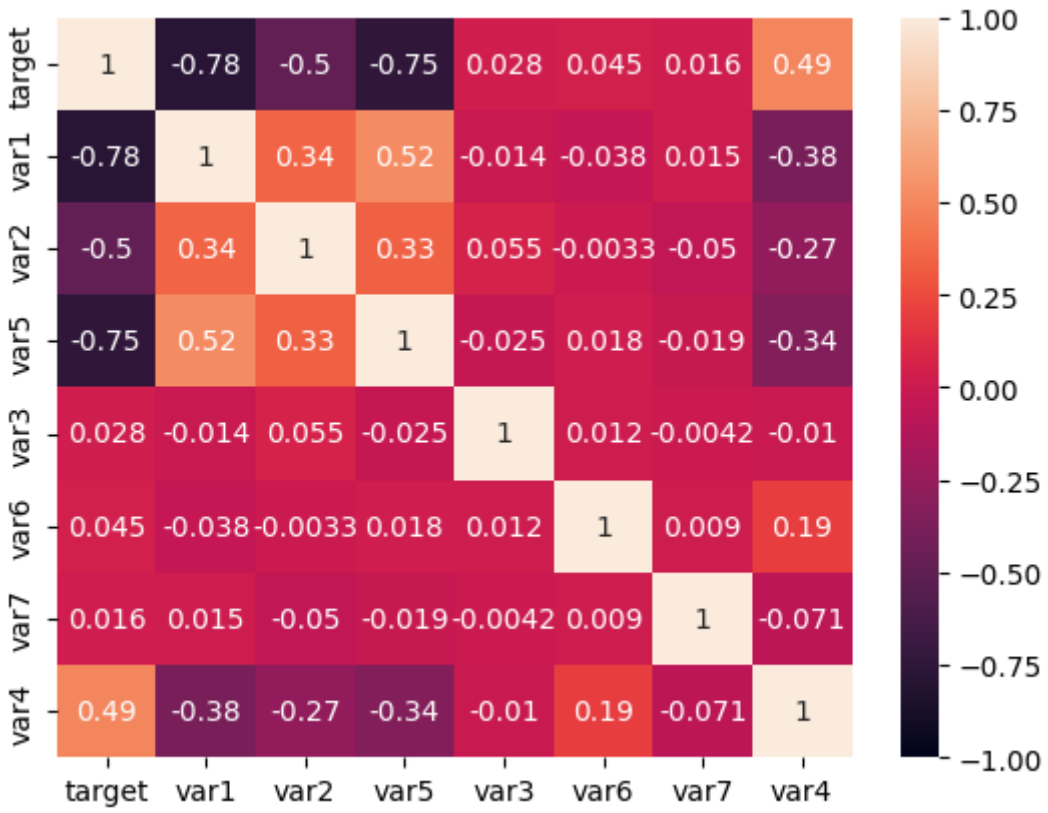
Without PCA all models showed close results to each other.
However, with PCA it is better to use KNN model. In this case we have non linear data. Moreover, PCA helps to remove correlated features and to get the most relevant features. And this is important for KNN as it sensitive for irrelevant features. Therefore, **PCA improved KNN model**.

While Logistic Regression make assumption about linearity between variables and Naive Bayes model make assumption about the features' independence, KNN have no assumptions about dataset.

Therefore, I decided that **the best model for this task is KNN**.

▼ Features (Answers to the questions 2.2.2 and 2.2.3)

To check the most critical features I plot a Spearman’s Correlation:



According to this plot - the most critical features for target are $var1$, $var2$, and $var5$. Since the absolute values of their correlation coefficients are the largest.

In addition, $var3$, $var6$, and $var7$ might be redundant.

▼ Additional research

$a)$ Multi-label learning problem - problem with 2 and more classes when we need to classify data that may belong to all classes or to no one of the classes.

$b)$ In initial problem we have two distinct classes and classified data belongs only to this classes.

If we add another column that will not exclude the $target$ values (for example, a column that will represent whether the temperature was above a certain value) - the given problem will transform into a multi-label problem.

In this case, my models will not work. Therefore, we will need to transform our problem into two Binary Classification problems to use previously implemented models. Or we can use another ML model.