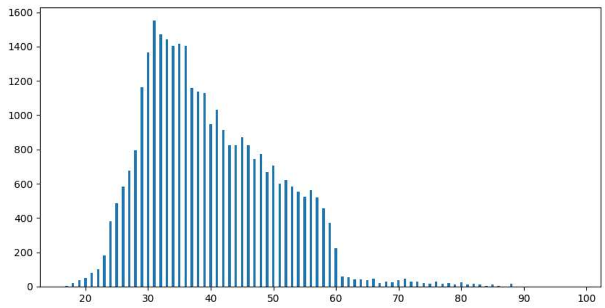
Daiichi – Coding Challenge
Documentation

<u>Approach:</u>

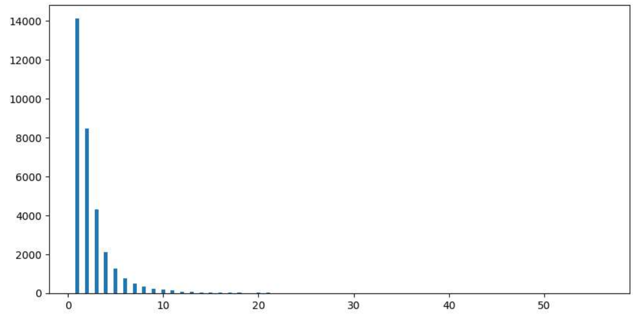
I've completed the Daiichi-Sankyo coding challenge by doing the following steps

Step 1: Printing value counts for all columns within the training dataset (also in graphics folder)
Step 2: Gather first insights from Step 1
Step 3: Correlation between features (also in graphics folder) → Answer question 1
Step 4: Feature importance (also in graphics folder) → Answer question 3
Step 5: Modelling approach
Step 6: Predictions (also in output folder)
Step 7: ToDos

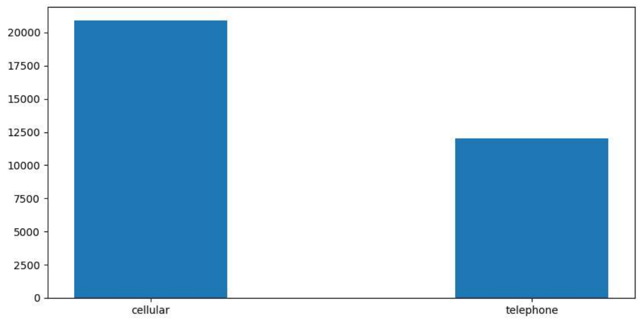
Step 1



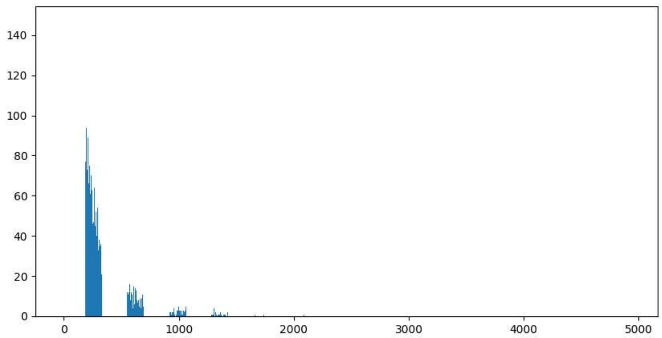
	age
count	32910.0
mean	40.01
std	10.40
min	17.0
25%	32.0
50%	38.0
75%	47.0
max	98.0



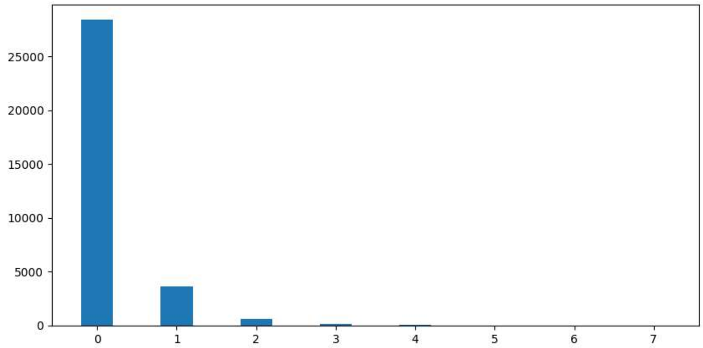
	campaign
count	32910.0
mean	2.56
std	2.75
min	1.0
25%	1.0
50%	2.0
75%	3.0
max	56.0



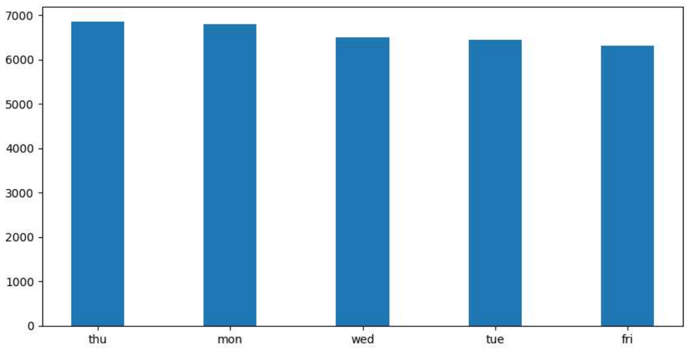
	contact
count	32910.0
cellular	20890
telephone	12020



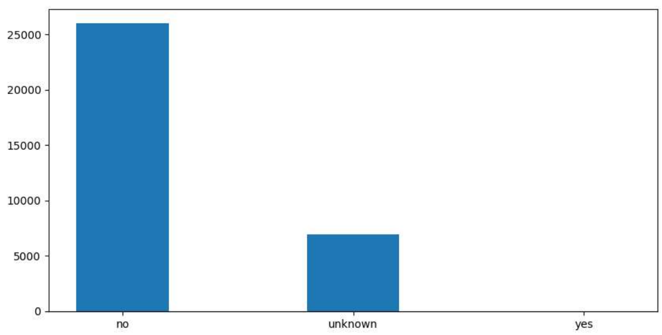
	duration
count	32910.0
mean	258.16
std	259.07
min	0.0
25%	103.0
50%	180.0
75%	319.0
max	4918.0



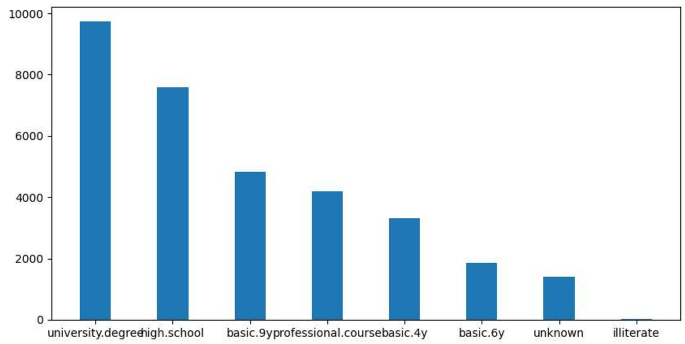
	previous
count	32910.0
mean	0.17
std	0.49
min	0.0
25%	0.0
50%	0.0
75%	0.0
max	7.0



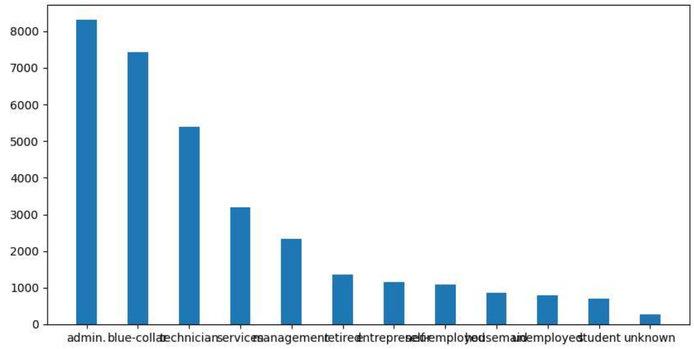
	day_of_week
count	32910.0
mon	6802
tue	6439
wed	6508
thu	6849
fri	6312



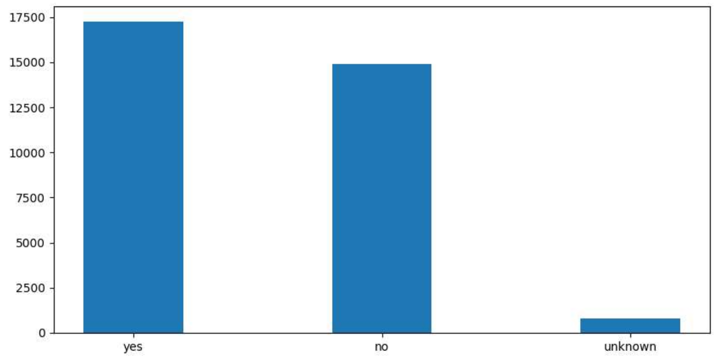
	default
count	32910.0
yes	3
no	25975
unknown	6932



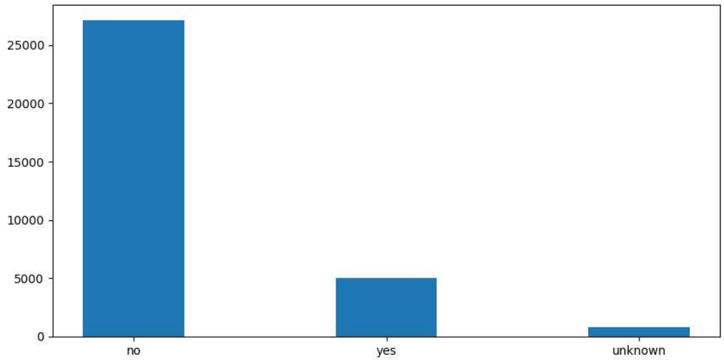
	education
count	32910.0
University.degree	9727
High.school	7585
Basic.9y	4818
Professional.course	4184
Basic.4y	3322
Basic.6y	1863
unknown	1395
illiterate	16



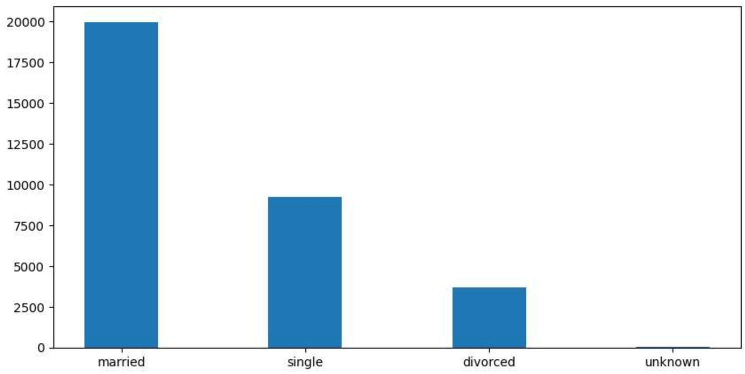
	job
count	32910.0
Admin.	8305
blue-collar	7430
technician	5392
service	3192
management	2343
retired	1364
entrepreneur	1159
self-employed	1098
housemaid	855
unemployed	798
student	710
unknown	264



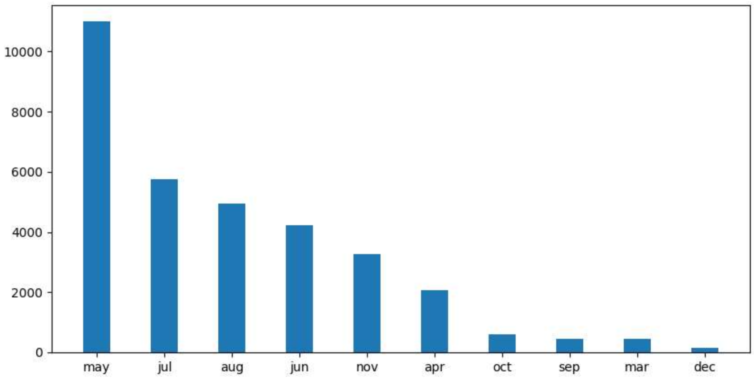
	housing
count	32910.0
yes	17236
no	14879
unknown	795



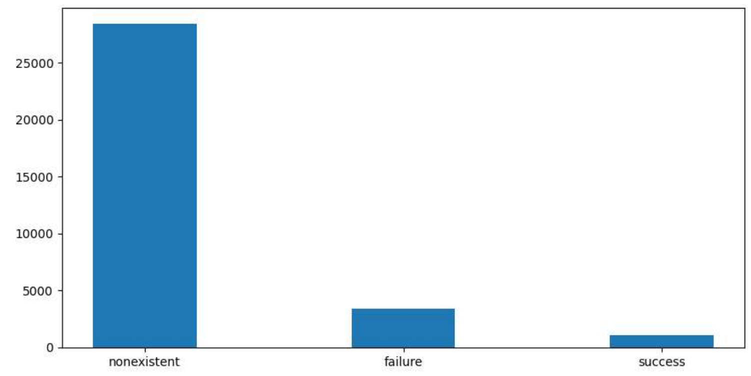
	loan
count	32910.0
yes	5016
no	27099
unknown	795



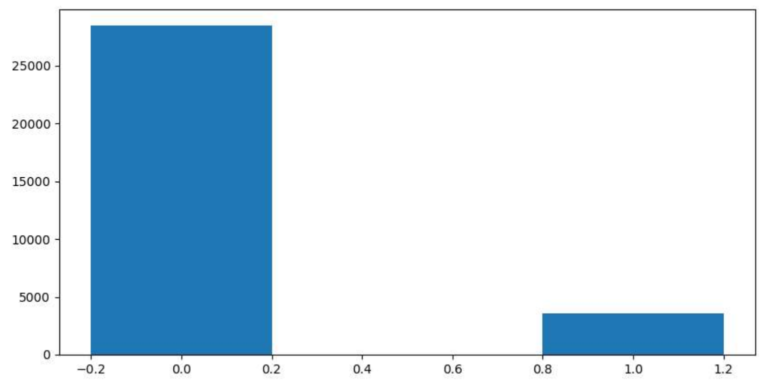
	marital
count	32910.0
married	19929
single	9245
unknown	65
divorced	3671



	job
count	32910.0
may	10993
jul	5753
aug	4946
jun	4242
nov	3263
apr	2083
oct	587
sep	464
mar	436
dec	143



	poutcome
count	32910.0
nonexistent	28280
failure	3426
success	1104



	y
count	32910.0
no	29203
yes	3707

Step 2

Insights:
- Training set is unbalanced → oversampling needed

- Many categorical features → one-hot encoding

- Marketing has a high focusing on people that:
- are in their 30s
- are married
- have no loans
- have high education
- have well paying jobs
- and have not been called in the past

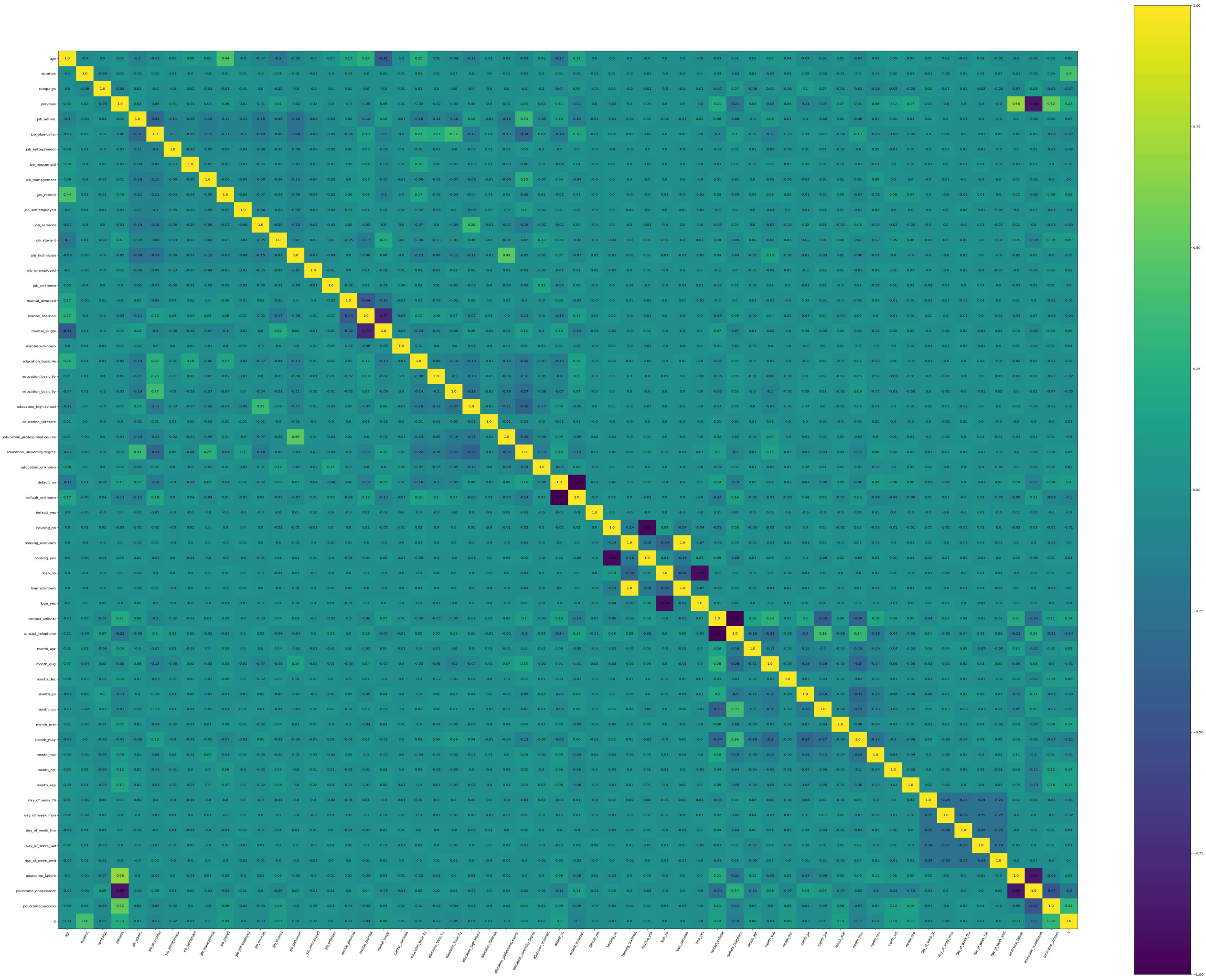
- Marketing team often calls between May and August. Seems to be best time to sell bank term deposits.

- marital, loan and housing have only few unknowns. These values can be dropped if they make up for less then 5% of the total rows in the dataset.

- The values in the duration column are in certain periods that are more or less 1 year appart.
→ assumption: Marketing team calls people during a certain period and offer term deposits that end ona specific date every year.

<u>Step 3</u>

The correlation-matrix for all features after one-hot-encoding the categorical features is rather large. A higher quality image of the matrix can be found in the graphics folder.



Answer question 1:

We see some correlations between variables.

1. Previous outcomes are correlated to the number of previous attempts. Because only if there was at least 1 previous attempt, the previous outcome can be a failure or a success.

2. Being retired is correlated to the age of a person. We see the same for people being married or divorced but less significant.

3. Some jobs and educations are correlated
- People with university degree are often admins or work in management
- People who visited a professional course are often technicians
- People with high scholl degrees often work in service
- People with basic 9, 6 or 4 year education often work a blue-collar job

4. Unknown housing is fully correlated with unknown loan

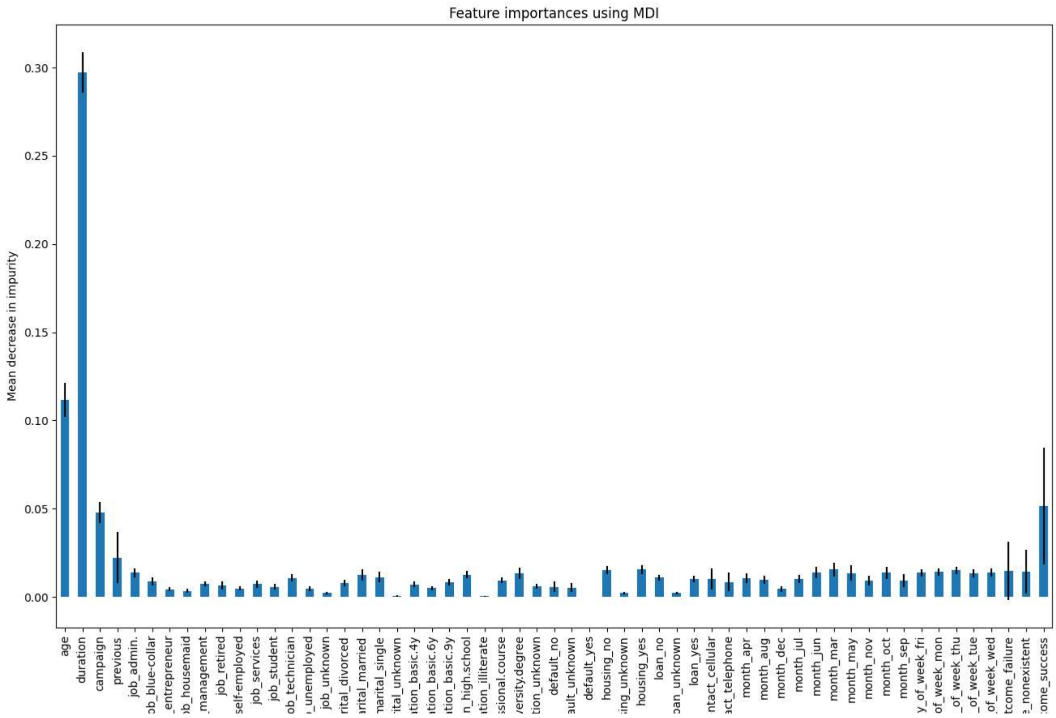
5. There is a slight correlation between people being contacted via telephone and people being contacted in the month of may and june.

6. Also people contacted via telephone have more often a non existent previous outcome, maybe because marketing could not reach them. Peolpe contacted by cellphone have more often a previous outcome that’s either success or failure.

7. The target y is mainly correlated to duration, previous attempts and if previous attempts were successfull. Some correlation also exists between y and people contacted via cellphone, and in some specific months (March, October, September). More about most important features in the next Step.

Step 4

Most important features for classification.



Answer question 3:

I used a random forest classifier to evaluate the improtance of features for the classification. Results show that age, duration, campaign. poutcome_success and previous have most influence on the classification results. Some of them already showed some correlation with the target in the correlation matrix.

Step 5

Answer question 2:

For the modelling I used a grid-search approach to find the best hyperparameters for my list of classification models. I evaluate the models based on the recall because I think it is most important to classify a success correcly.

Before the grid-search I perform the following preprocessing steps:

1. Remove unknown values as long as the total of removed lines is not more than 5% of the total rows of the dataset. This will remove some features created by one-hot-encoding but when we look into the feature importance plot in Step 4 we see that these features are very unimportant for the classification. Ignoring them should be fine.

2. Make the target binary (1,0 instead of yes, no)
3. One-hot-encoding of categorical features
4. Balance dataset with SMOTE oversampling. We could also duplicate the rows of the minority class but it would lead a less generalized model.
5. Run a standardscaler to make the features have mean 0 and standard deviation 1. This improves the performance of some models (e.g. neural networks)
6. Run principle component analysis (PCA) to reduce the dimensions of the classification problem. Alternatively we could also just remove features with low importance

Some results:

1. Train run without SMOTE oversampling:

estimator	**mean_test_accuracy**	**mean_test_recall**	**mean_test_f1_score**
MLPClassifier(hidden_layer_sizes=(64, 32, 2), learning_rate='adaptive', random_state=0, solver='lbfgs'), (64, 32, 2), 15	0.8735946520814343	0.47991081762159943	0.46096822841719315

The MLP Classifier gives the best train results without oversampling. We can see that accuracy is ok but recall is rather low. This is expected because the classifiers „sees“ way more samples of the majority class during training. As a result he tend to predict the majority class because it’s most of the time correct during training. In the test the model „ignores“ the minority class what leads to low recall in this case.

2. Train run with SMOTE oversampling:

estimator	**mean_test_accuracy**	**mean_test_recall**	**mean_test_f1_score**
KNeighborsClassifier(n_neighbors=3), 15	0.9258131986179509	0.8921308026161535	0.9163780779362591

After oversampling the dataset with SMOTE the best classifier is the Kneighbors Classifier. We can see that the recall is significantly better. Also accuracy and f1 score are high enough to accept the model.

3. Train run with removed unknowns from loan, housing, and marital:

estimator	**mean_test_accuracy**	**mean_test_recall**	**mean_test_f1_score**
KNeighborsClassifier(n_neighbors=3), 15	0.9265615958079594	0.896048572016089	0.9178202646030572

Removing unknowns from columns with only few unknown doesn’t affect the model choice but improves the errors again a little bit.

Step 6

The prediction results with the model from the 3. train run classifies the unknown test data (test_file.xlsx) and classifies exactly two datapoint as success.

Please find the results in output/predictions.csv

Step 7

Possible next steps would be to:
- Collect more data :)
- Add more classiffiers and hyperparameter ranges to the gridsearch
- Add functionality to not overwite a already trained model with another one that has lower recall
- Add a pre-commit hock to ensure high code quality in the repo
- Add pipeline yaml files to deploy the code in MS Azure
- Add mlflow to track experiments