Pożyczki Konsumenckie

Paweł Pozorski

Michał Pytel

Dane

#	Column	Non-Null Count	Dtype
0	PRODUCT	15097 non-null	object
1	AGE	15097 non-null	int64
2	AREA	15097 non-null	object
3	RESIDENTIAL_PLACE	15097 non-null	object
4	EDUCATION	15097 non-null	object
5	MARITAL_STATUS	15097 non-null	object
6	HOUSEHOLD_MEMBERS	15097 non-null	int64
7	NO_OF_DEPENDENTS	15097 non-null	int64
8	INCOME	15097 non-null	float64
9	WORK_SENIORITY	15097 non-null	int64
10	BUSINESS AGE	15097 non-null	int64
11	ECONOMIC_SECTOR	15097 non-null	object
12	EMPLOYEE_NO	15097 non-null	object
13	LENGTH_RELATIONSHIP_WITH_CLIENT	15097 non-null	int64
14	DEBIT_CARD	15097 non-null	int64
15	CURRENT_ACCOUNT	15097 non-null	int64
16	SAVING_ACCOUNT	15097 non-null	int64
17	SALARY_ACCOUNT	15097 non-null	int64
18	FOREIGN_ACCOUNT	15097 non-null	int64
19	FINALIZED_LOAN	15097 non-null	int64
20	DEPOSIT	15097 non-null	int64
21	PENSION_FUNDS	15097 non-null	int64

	dtype	missing	example_row_1	example_row_2	example_row_3	example_row_4
PRODUCT	object	0	C	C	F	C
AGE	int64	0	65	64	30	39
AREA	object	0	County capital	County capital	Urban area	County capital
RESIDENTIAL_PLACE	object	0	Owner without mortgage	Owner without mortgage	Living with family	Owner without mortgage
EDUCATION	object	0	University	University	University	Post-graduate
MARITAL_STATUS	object	0	married	married	married	divorced
HOUSEHOLD_MEMBERS	int64	0	2	2	2	1
NO_OF_DEPENDENTS	int64	0	0	0	0	0
INCOME	float64	0	1245.0	1380.0	1131.0	1730.0
WORK_SENIORITY	int64	0	5	5	2	9
BUSINESS AGE	int64	0	16	16	6	13
ECONOMIC_SECTOR	object	0	Missing	Missing	Other	Education
EMPLOYEE_NO	object	0	Missing	Missing	> 1.000	between 11-20
LENGTH_RELATIONSHIP_WITH_CLIENT	int64	0	1	8	1	2
DEBIT_CARD	int64	0	0	0	1	0
CURRENT_ACCOUNT	int64	0	0	0	1	0
SAVING_ACCOUNT	int64	0	0	0	0	0
SALARY_ACCOUNT	int64	0	0	0	0	0
FOREIGN_ACCOUNT	int64	0	0	0	0	0
FINALIZED_LOAN	int64	0	0	0	0	0
DEPOSIT	int64	0	0	0	0	0
PENSION_FUNDS	int64	0	0	0	0	0

Dla kogo jest ten model?

Dla kogo jest ten model?

Dla Pożyczkobiorców

Dlaczego to może być przydatne?

Dlaczego to może być przydatne dla pożyczkobiorców?

- decyzje przyznania pożyczki

Dlaczego to może być przydatne dla pożyczkobiorców?

- decyzje przyznania pożyczki
- Wpływ cech

Dlaczego to może być przydatne dla pożyczkobiorców?

- decyzje przyznania pożyczki
- Wpływ cech
- Co można polepszyć, aby dostać pożyczkę

Dlaczego to może być przydatne dla pożyczkobiorców?

- decyzje przyznania pożyczki
- Wpływ cech
- Co można polepszyć, aby dostać pożyczkę
- Oszczędność czasu po stronie konsumenta i banku

Czy można to skomercjalizować?

Czy można to skomercjalizować?

TAK

Na jakiej zasadzie to działa?

- Podajemy cechy pożyczkobiorcy

Na jakiej zasadzie to działa?

- Podajemy cechy pożyczkobiorcy
- Na podstawie cech model podejmuje decyzje

Na jakiej zasadzie to działa?

- Podajemy cechy pożyczkobiorcy
- Na podstawie cech model podejmuje decyzje
- Możliwe wyniki: 1 lub 0

Na jakiej zasadzie to działa?

- Podajemy cechy pożyczkobiorcy
- Na podstawie cech model podejmuje decyzje
- Możliwe wyniki: 1 lub 0
- Finalny model pozwala również na zwrócenie wyznaczonego prawdopodobienstwa otrzymania kredytu – finalna decyzja pozostawiona klientowi

Pipeline

- Wykorzystaliśmy pipeline do przeróbki danych pod optymalne działanie modelu

Pipeline

- Wykorzystaliśmy pipeline do przeróbki danych pod optymalne działanie modelu
- Te pipeliny usuwają mało znaczące cechy

Pipeline

- Wykorzystaliśmy pipeline do przeróbki danych pod optymalne działanie modelu
- Te pipeliny usuwają mało znaczące cechy
- Standaryzują dane

Pipeline

- Wykorzystaliśmy pipeline do przeróbki danych pod optymalne działanie modelu
- Te pipeliny usuwają mało znaczące cechy
- Standaryzują dane
- Uzupełniają missing values

Pipeline

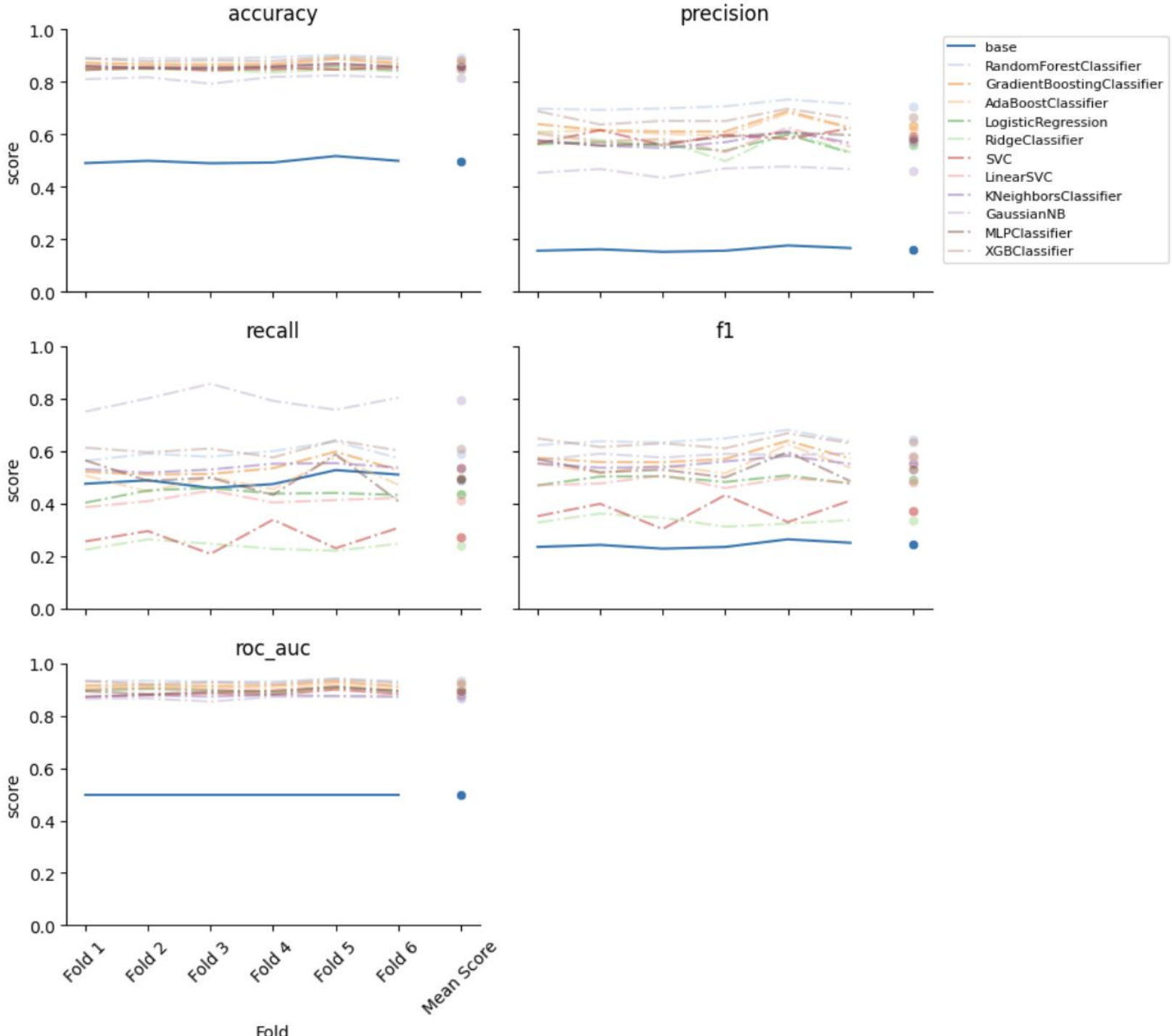
- Wykorzystaliśmy pipeline do przeróbki danych pod optymalne działanie modelu
- Te pipeliny usuwają mało znaczące cechy
- Standaryzują dane
- Uzupełniają missing values
- Wszystko po to aby uławtić działanie użytkowników naszego rozwiązania

Pipeline

Czyli zawężają interfejs użytkownika do 2 komend – predict() i predict_proba() + wczytanie go do ramu.

Jaki Model?

Wstępne poszukiwania



Wstępne poszukiwania

Models	roc_auc
RandomForestClassifier	0.9329136509618493
XGBClassifier	0.9271466976381806
GradientBoostingClassifier	0.9167174808016347
AdaBoostClassifier	0.9088979595380599
LinearSVC	0.8995210164868662
LogisticRegression	0.8981257017506247
RidgeClassifier	0.89739705341036
MLPClassifier	0.8938190855011617
SVC	0.8831598680627386
KNeighborsClassifier	0.8749405763990152
GaussianNB	0.8670563180878653

Wstępne poszukiwania

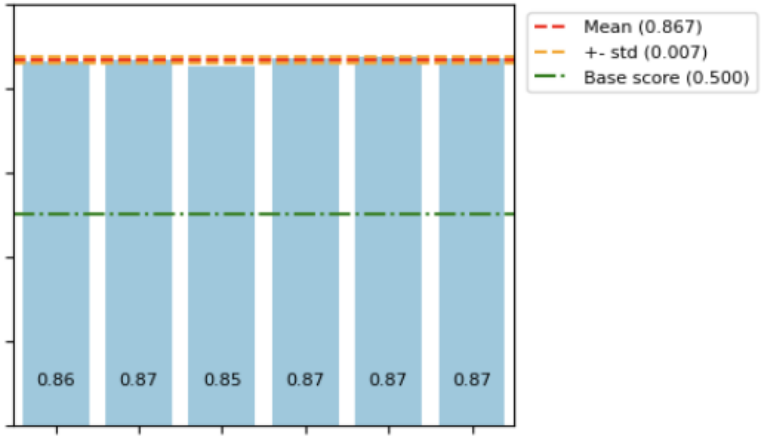
Models	f1
RandomForestClassifier	0.643612386009729
XGBClassifier	0.634054002788925
GaussianNB	0.583110778603261
GradientBoostingClassifier	0.578826885256998
KNeighborsClassifier	0.553935585093570
AdaBoostClassifier	0.549288137377603
MLPClassifier	0.533610524788320
LogisticRegression	0.490692980616519
LinearSVC	0.481916546274774
SVC	0.371525478023150
RidgeClassifier	0.334876190595438

Wstępne poszukiwania

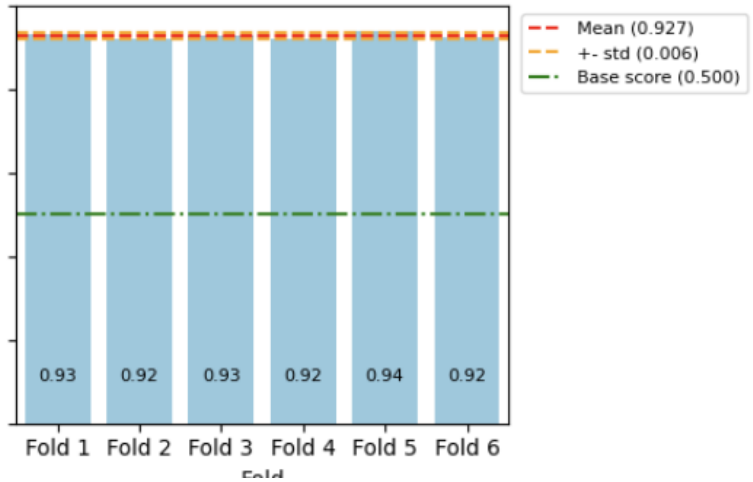
Models	recall
GaussianNB	0.7938857111664775
XGBClassifier	0.6066934921024825
RandomForestClassifier	0.5905719120531207
KNeighborsClassifier	0.537315818819134 8
GradientBoostingClassifier	0.5344948591079762
MLPClassifier	0.496947047057585
AdaBoostClassifier	0.4933384800739258 3
LogisticRegression	0.437690322178163
LinearSVC	0.4142911729499791 4
SVC	0.272697320965559 7
RidgeClassifier	0.2384081365289913 7

Zbadajmy kandydatów

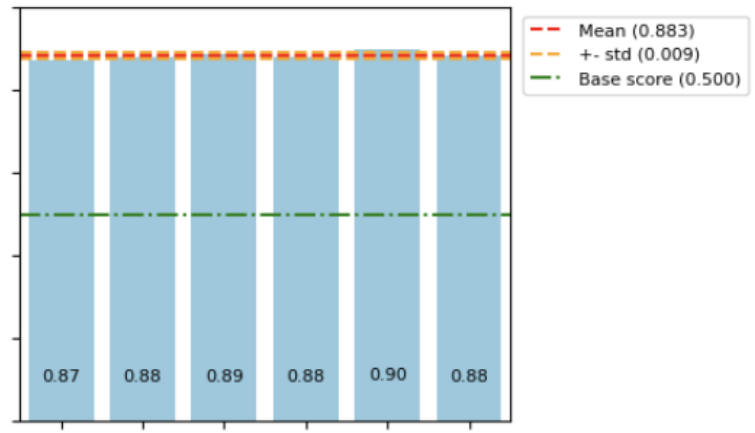
Kandydaci



GaussianNB

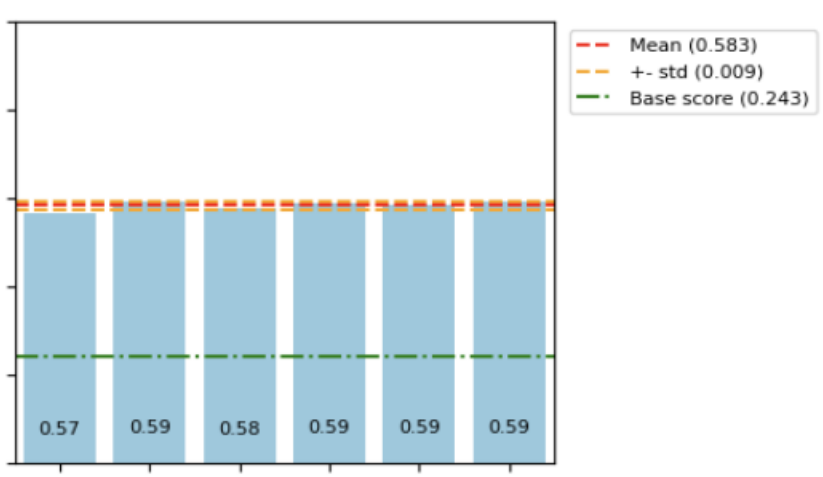


XGBClassifier

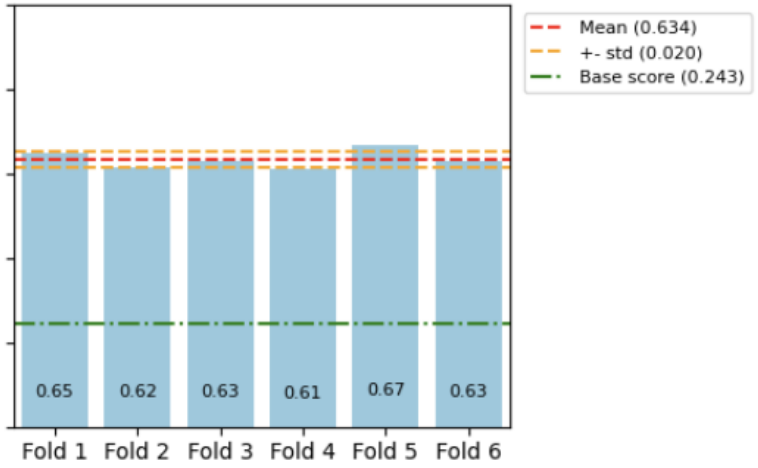


SVC

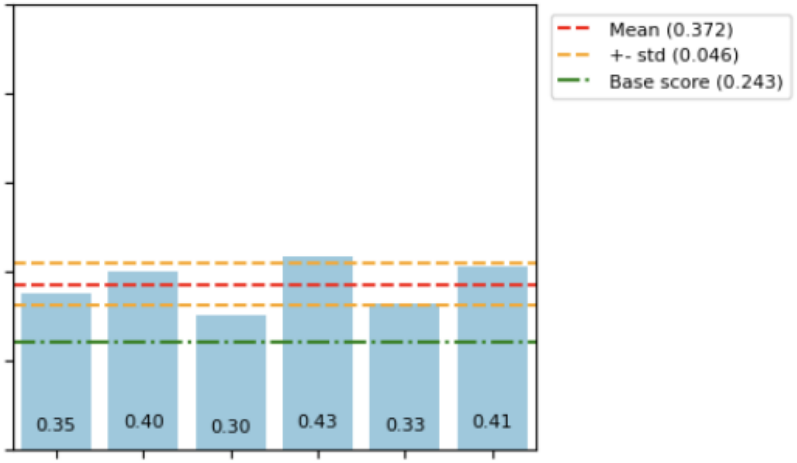
Kandydaci



GaussianNB

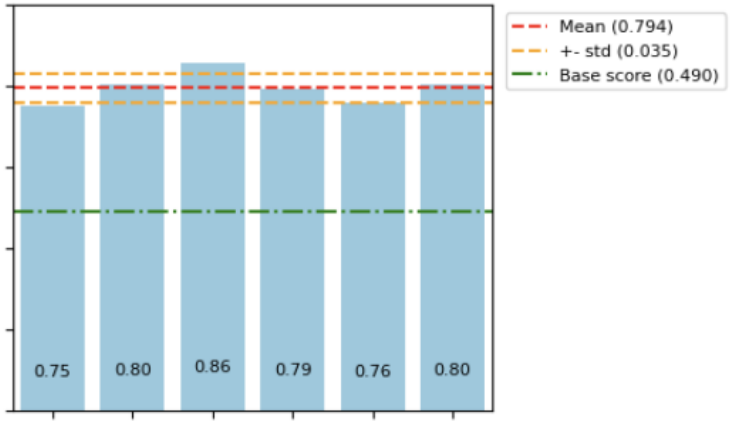


XGBClassifier

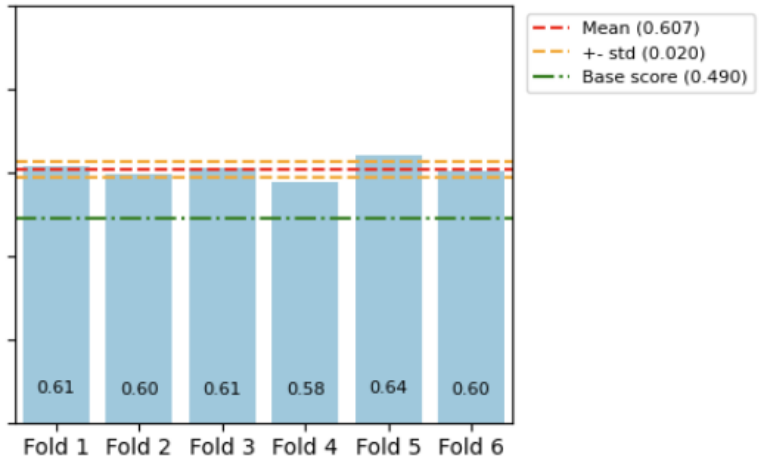


SVC

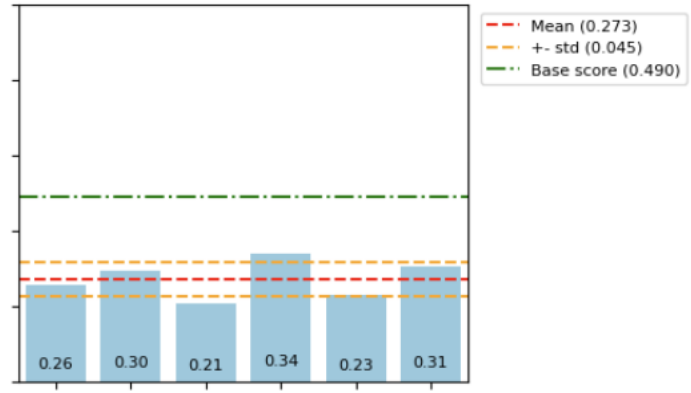
Kandydaci



GaussianNB



XGBClassifier



SVC

XGBClassifier

	precision	recall	f1-score	support
0	0.93	0.94	0.94	2734
1	0.66	0.60	0.63	501
accuracy			0.89	3235
macro avg	0.79	0.77	0.78	3235
weighted avg	0.89	0.89	0.89	3235

ROC AUC score: 0.7730504608924068

```
Number of finished trials: 100
Best trial:
  Value: 0.8907731168383899
  Params:
    booster: dart
    lambda: 5.300729413321117
    alpha: 0.004373402847043272
    max_depth: 46
    eta: 0.36324883963950516
    gamma: 0.0014298698374813413
    grow_policy: depthwise
Refitted best model f1-score on valid: 0.8905718701700155
```

SVC

	precision	recall	f1-score	support
0	0.90	0.96	0.93	2734
1	0.68	0.44	0.53	501
accuracy			0.88	3235
macro avg	0.79	0.70	0.73	3235
weighted avg	0.87	0.88	0.87	3235

ROC AUC score: 0.7005411269633374

```
Number of finished trials: 40
Best trial:
  Value: 0.8767968868249433
  Params:
    C: 22208.56815131227
    kernel: rbf
    max_iter: 5000
    probability: True
    gamma: 0.26672481100997353
Refitted best model f1-score on valid: 0.8809891808346213
```

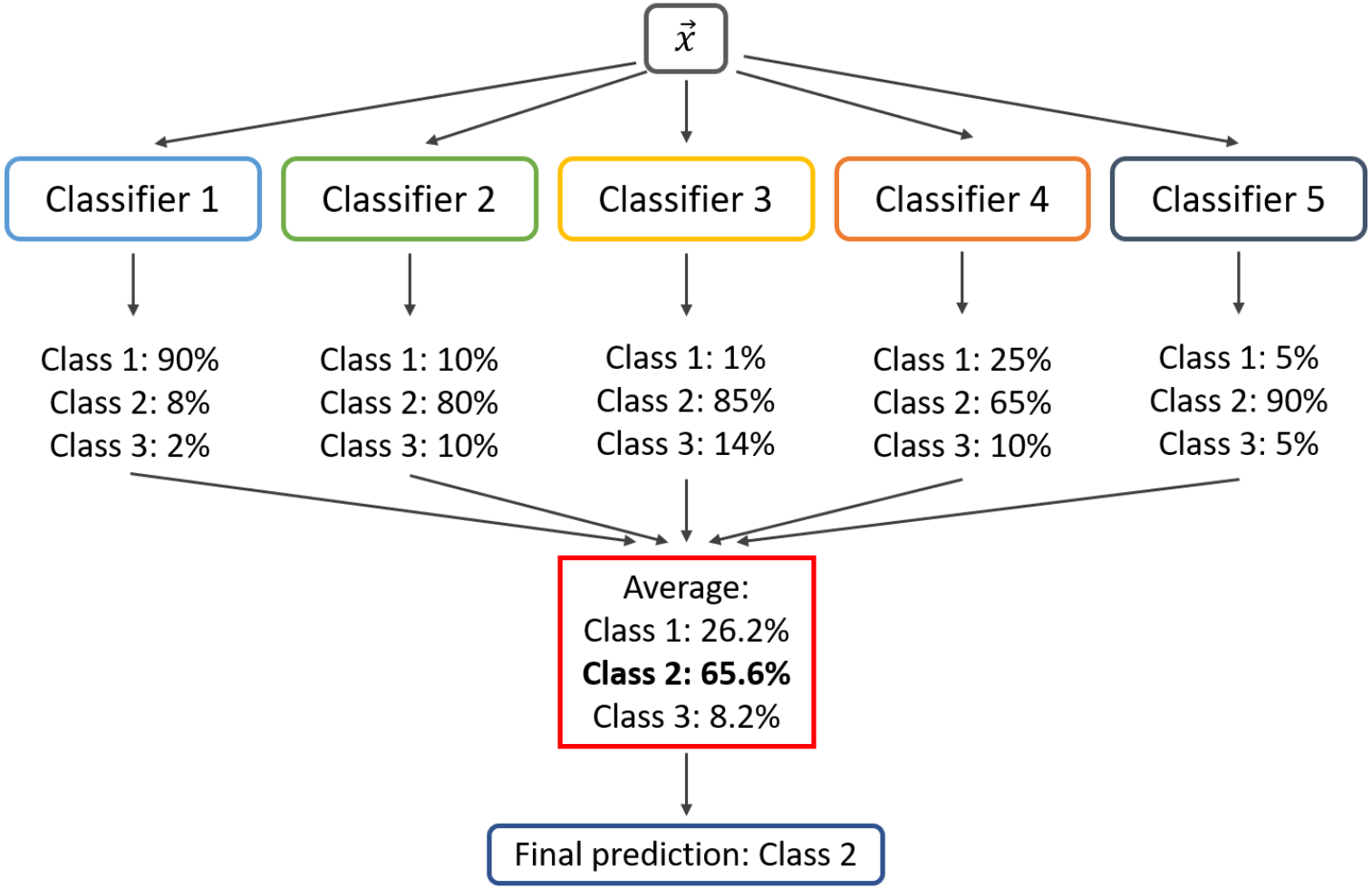
GaussianNB

	precision	recall	f1-score	support
0	0.97	0.82	0.89	2734
1	0.47	0.85	0.60	501
accuracy			0.83	3235
macro avg	0.72	0.84	0.75	3235
weighted avg	0.89	0.83	0.85	3235

ROC AUC score: 0.8361853469359743

```
Number of finished trials: 100
Best trial:
  Value: 0.892736429389340 8
  Params:
    var_smoothing: 9.026696330003137e-05
Refitted best model f1-score on valid: 0.8278207109737249
```

Voting Classifier



Voting Classifier

	precision	recall	f1-score	support
0	0.94	0.92	0.93	2734
1	0.61	0.69	0.65	501
accuracy			0.88	3235
macro avg	0.78	0.80	0.79	3235
weighted avg	0.89	0.88	0.89	3235

ROC AUC score: 0.8046259346705272

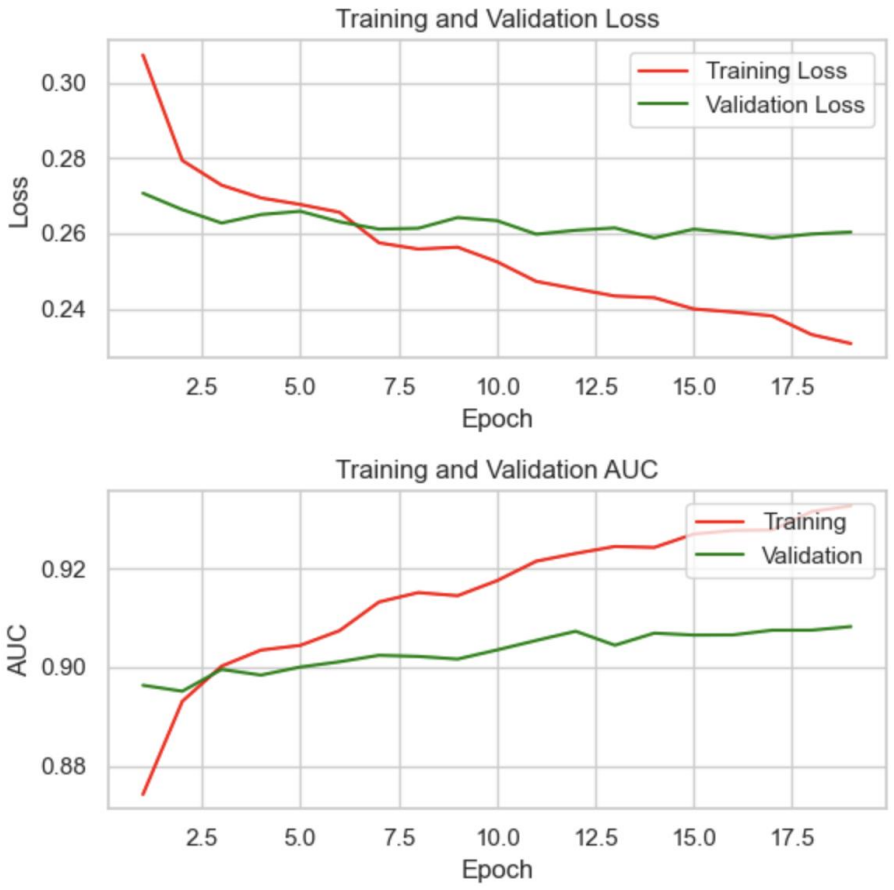
Let's go deeper

Deep learning

SCORES

Name	auc	f1_score
Simple Classifier (Dropout=0.2)	0.693110852	0.503954802
Simple Classifier (Dropout=0.0)	0.696272050	0.506666667
Simple Classifier (Dropout=0.5)	0.643539914	0.421319797
Residual Net	0.500000000	0.268201285
Drop Connect Net	0.662005543	0.451306413
Dense Net	0.684860345	0.490825688

Simple Classifier



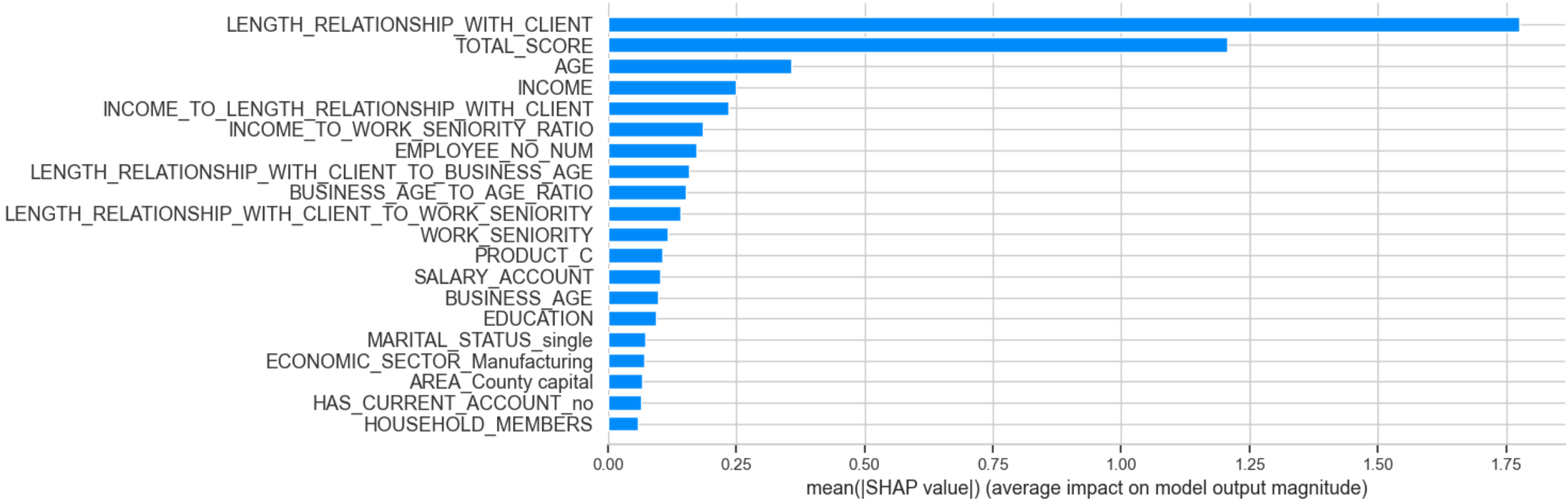
Layer (type)	Output Shape	Param #
dense (Dense)	(None, 256)	14,080
leaky_re_lu (LeakyReLU)	(None, 256)	0
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 128)	32,896
leaky_re_lu_1 (LeakyReLU)	(None, 128)	0
dense_2 (Dense)	(None, 128)	16,512
leaky_re_lu_2 (LeakyReLU)	(None, 128)	0
dense_3 (Dense)	(None, 64)	8,256
leaky_re_lu_3 (LeakyReLU)	(None, 64)	0
dropout_1 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 1)	65

Simple Classifier

	precision	recall	f1-score	support
0	0.90	0.94	0.92	2734
1	0.58	0.45	0.50	501
accuracy			0.86	3235
macro avg	0.74	0.69	0.71	3235
weighted avg	0.85	0.86	0.86	3235

ROC AUC score: 0.6931108521800583

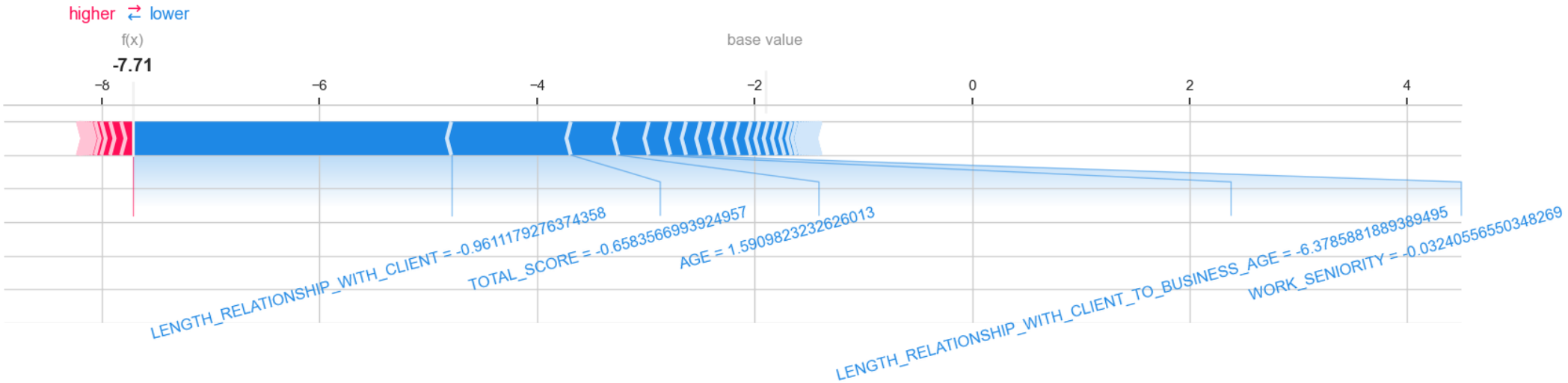
Jak to działa?

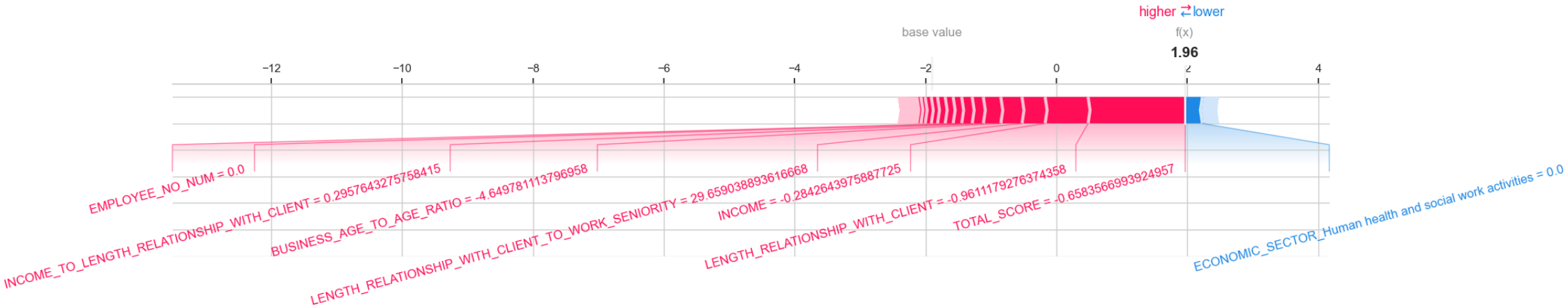

LENGTH_RELATIONSHIP_WITH_CLIENT
TOTAL_SCORE
AGE
INCOME
INCOME_TO_LENGTH_RELATIONSHIP_WITH_CLIENT
INCOME_TO_WORK_SENIORITY_RATIO
EMPLOYEE_NO_NUM
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_BUSINESS_AGE
BUSINESS_AGE_TO_AGE_RATIO
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_WORK_SENIORITY
WORK_SENIORITY
PRODUCT_C
SALARY_ACCOUNT
BUSINESS_AGE
EDUCATION
MARITAL_STATUS_single
ECONOMIC_SECTOR_Manufacturing
AREA_County capital
HAS_CURRENT_ACCOUNT_no
HOUSEHOLD_MEMBERS
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
mean(|SHAP value|) (average impact on model output magnitude)

Jak zrobiliśmy kolumny dotworzone

```python
class FeatureCorrelationEngineer(BaseEstimator, TransformerMixin):
    def __init__(self, cols_to_combine, target_col, new_name=None, drop=False):
        super().__init__()
        self.cols_to_combine = cols_to_combine
        self.target_col = target_col
        self.weights = np.ones(len(cols_to_combine))
        self.new_name = new_name if new_name is not None else "_".join(cols_to_combine)
        self.drop = drop
        self.scaler = StandardScaler()

    def get_combined_value(self, X):
        return X[self.cols_to_combine].values.dot(self.weights.reshape(-1, 1))

    def get_corr(self, X):
        return np.corrcoef(self.get_combined_value(X).ravel(), X[self.target_col])[0, 1]

    def fit(self, X, y=None):
        def get_score(weights):
            self.weights = weights
            return -np.abs(self.get_corr(X))

        self.weights = minimize(get_score, self.weights, method="Nelder-Mead")["x"]

        new_col = self.get_combined_value(X)
        self.scaler.fit(new_col)

        return self

    def transform(self, X):
        new_col = self.get_combined_value(X)
        new_col = self.scaler.transform(new_col)
        X[self.new_name] = new_col

        if self.drop:
            X.drop(columns=self.cols_to_combine, inplace=True)
        return X

    def set_output(self, *args, **kwargs):
        return self
```

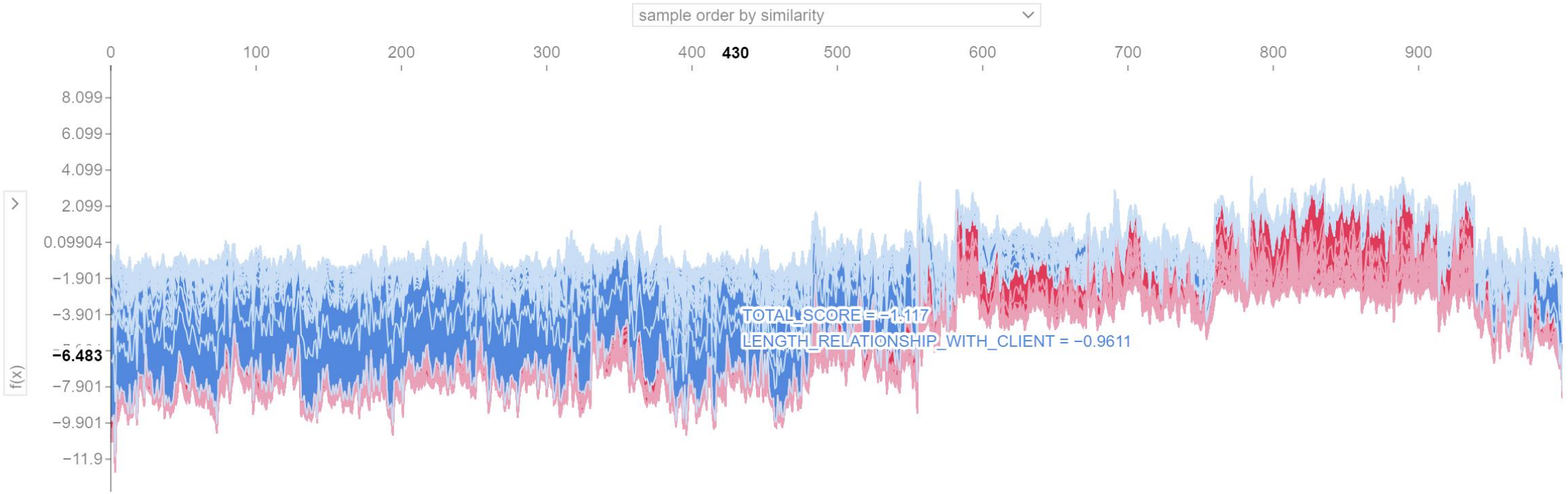
```python
class CreateAdditionalFeatures(BaseEstimator, TransformerMixin):
    def fit(self, X, y=None):
        return self

    def transform(self, X):
        # Calculate additional features
        X["LENGTH_RELATIONSHIP_WITH_CLIENT_TO_WORK_SENIORITY"] = (
            X["LENGTH_RELATIONSHIP_WITH_CLIENT"] / X["WORK_SENIORITY"]
        )
        X["INCOME_TO_WORK_SENIORITY_RATIO"] = X["INCOME"] / X["WORK_SENIORITY"]
        X["BUSINESS_AGE_TO_AGE_RATIO"] = X["BUSINESS_AGE"] / X["WORK_SENIORITY"]
        X["LENGTH_RELATIONSHIP_WITH_CLIENT_TO_BUSINESS_AGE"] = (
            X["LENGTH_RELATIONSHIP_WITH_CLIENT"] / X["BUSINESS_AGE"]
        )
        X["INCOME_TO_LENGTH_RELATIONSHIP_WITH_CLIENT"] = (
            X["INCOME"] / X["LENGTH_RELATIONSHIP_WITH_CLIENT"]
        )

        return X

    def set_output(self, *args, **kwargs):
        return self
```


higher ⇄ lower
f(x)
-7.71
base value
−8
−6
−4
−2
0
2
4
LENGTH_RELATIONSHIP_WITH_CLIENT = -0.9611179276374358
TOTAL_SCORE = -0.6583566993924957
AGE = 1.5909823232626013
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_BUSINESS_AGE = -6.3785881889389495
WORK_SENIORITY = -0.032405565503482 69


higher ⇄ lower
base value
f(x)
1.96
−12
−10
−8
−6
−4
−2
0
2
4
EMPLOYEE_NO_NUM = 0.0
INCOME_TO_LENGTH_RELATIONSHIP_WITH_CLIENT = 0.2957643275758415
BUSINESS_AGE_TO_AGE_RATIO = -4.649781113796958
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_WORK_SENIORITY = 29.659038893616668
INCOME = -0.2842643975887725
LENGTH_RELATIONSHIP_WITH_CLIENT = -0.9611179276374358
TOTAL_SCORE = -0.6583566993924957
ECONOMIC_SECTOR_Human health and social work activities = 0.0

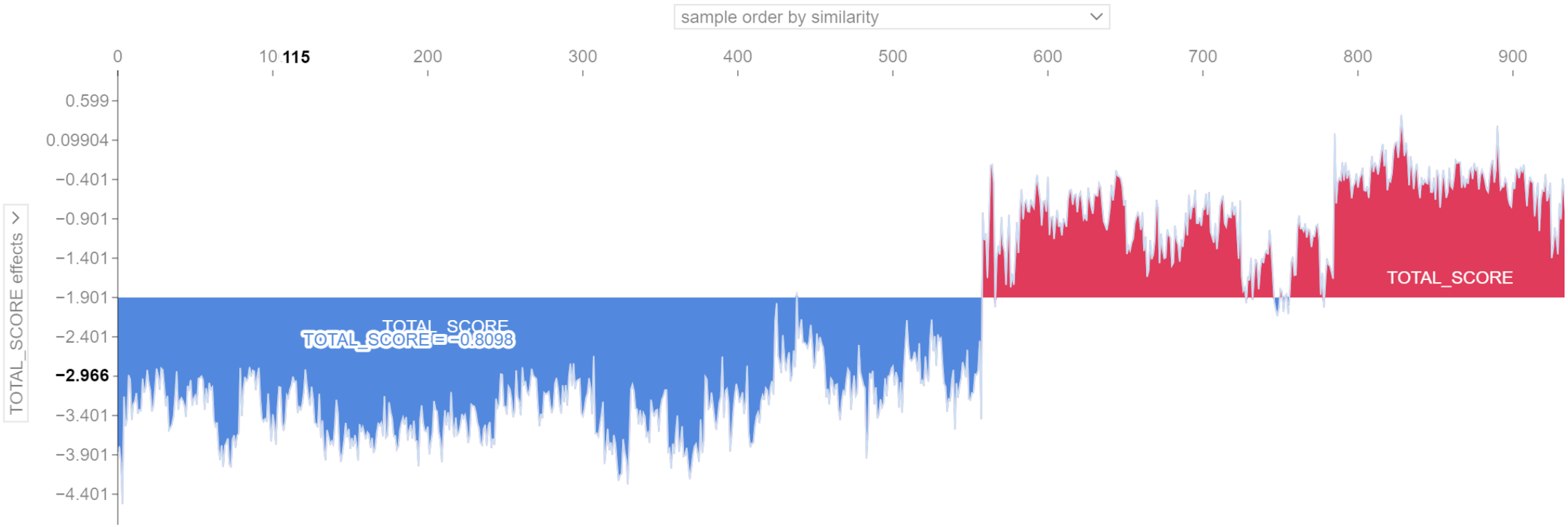
```
shap.force_plot(
    explainer.expected_value, shap_values[:1000, :], X_train.iloc[:1000, :]
)
```

[42] ✓ 1.7s Python



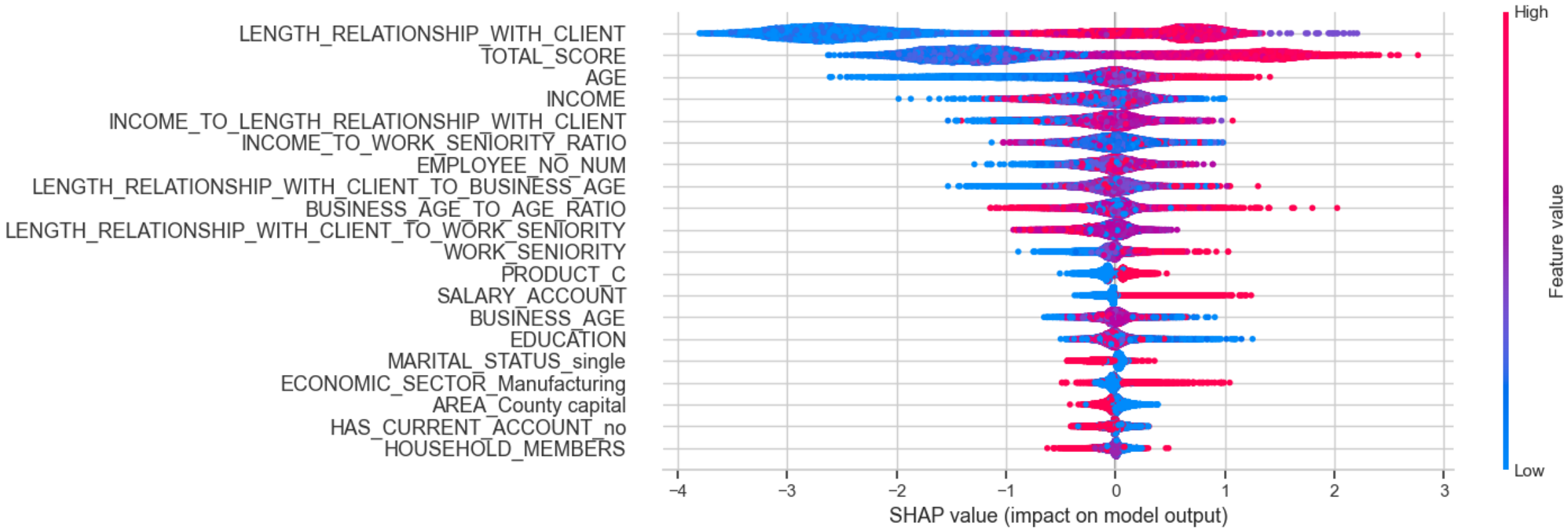
```
shap.force_plot(
    explainer.expected_value, shap_values[:1000, :], X_train.iloc[:1000, :]
)
```

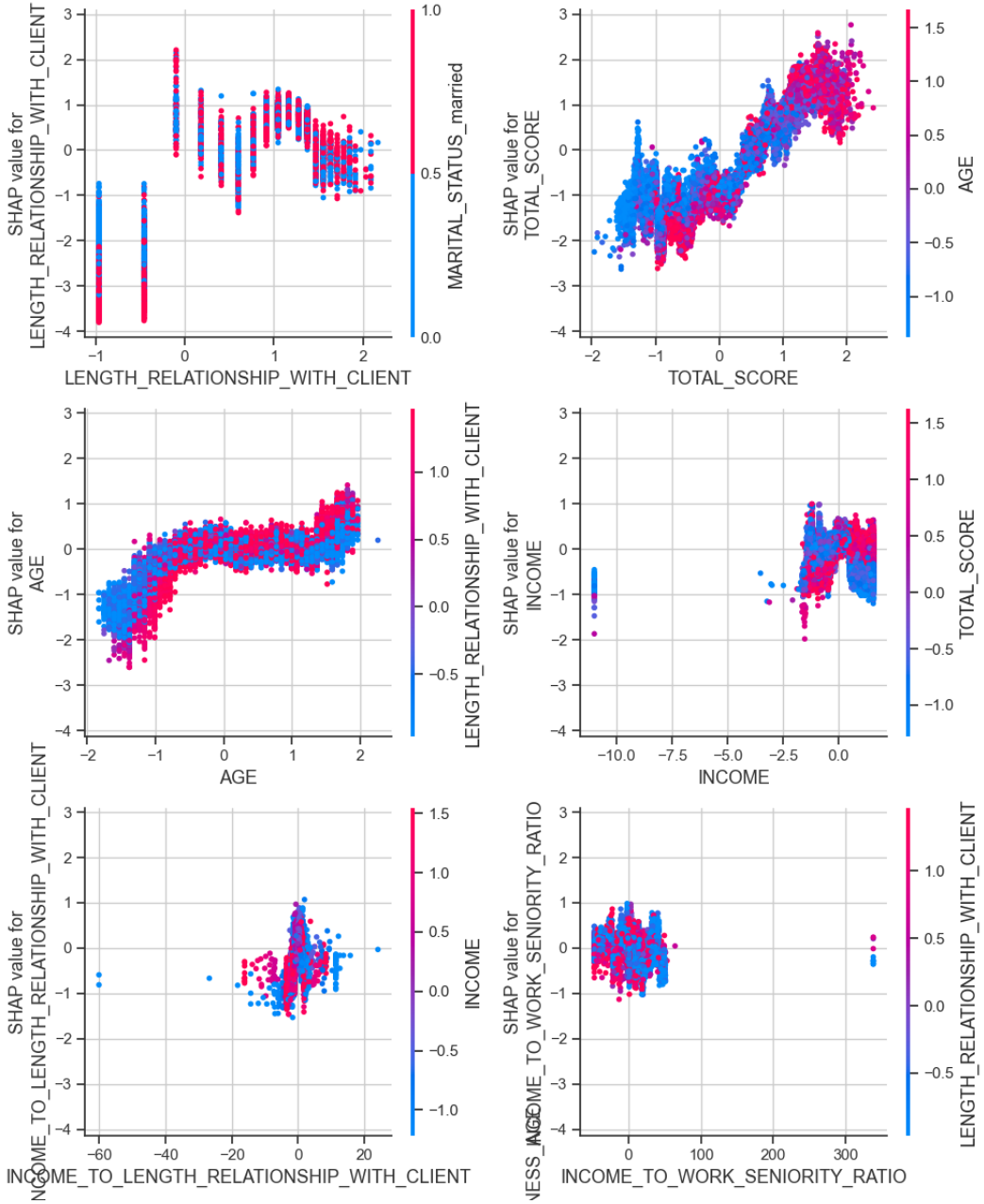
[42] ✓ 1.7s

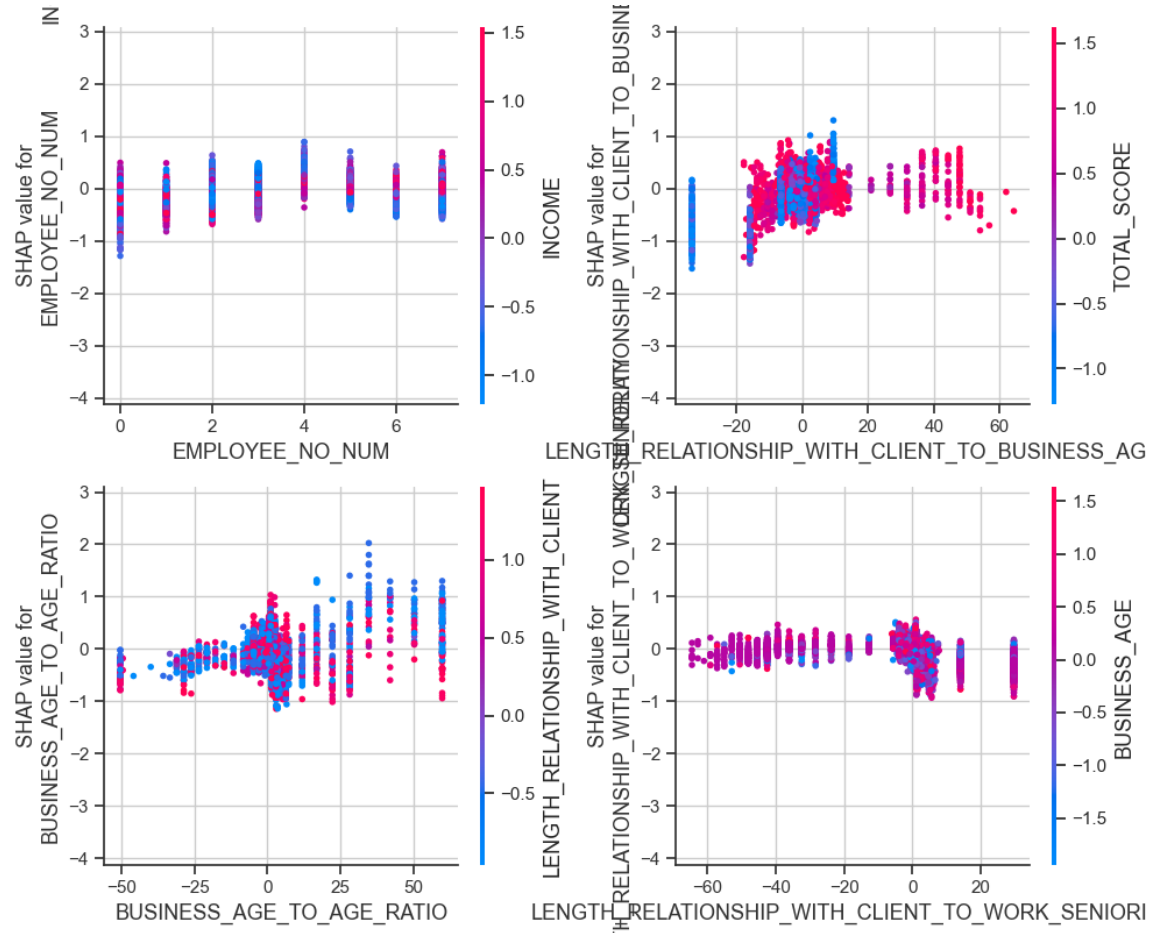


```
shap.force_plot(
    explainer.expected_value, shap_values[:1000, :], X_train.iloc[:1000, :]
)
```

[42] ✓ 1.7s Python


LENGTH_RELATIONSHIP_WITH_CLIENT
TOTAL_SCORE
AGE
INCOME
INCOME_TO_LENGTH_RELATIONSHIP_WITH_CLIENT
INCOME_TO_WORK_SENIORITY_RATIO
EMPLOYEE_NO_NUM
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_BUSINESS_AGE
BUSINESS_AGE_TO_AGE_RATIO
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_WORK_SENIORITY
WORK_SENIORITY
PRODUCT_C
SALARY_ACCOUNT
BUSINESS_AGE
EDUCATION
MARITAL_STATUS_single
ECONOMIC_SECTOR_Manufacturing
AREA_County capital
HAS_CURRENT_ACCOUNT_no
HOUSEHOLD_MEMBERS
−4
−3
−2
−1
0
1
2
3
SHAP value (impact on model output)
High
Feature value
Low


SHAP value for LENGTH_RELATIONSHIP_WITH_CLIENT
LENGTH_RELATIONSHIP_WITH_CLIENT
MARITAL_STATUS_married
SHAP value for TOTAL_SCORE
TOTAL_SCORE
AGE
SHAP value for AGE
AGE
LENGTH_RELATIONSHIP_WITH_CLIENT
SHAP value for INCOME
INCOME
TOTAL_SCORE
SHAP value for INCOME_TO_LENGTH_RELATIONSHIP_WITH_CLIENT
INCOME_TO_LENGTH_RELATIONSHIP_WITH_CLIENT
INCOME
SHAP value for INCOME_TO_WORK_SENIORITY_RATIO
INCOME_TO_WORK_SENIORITY_RATIO
LENGTH_RELATIONSHIP_WITH_CLIENT


SHAP value for EMPLOYEE_NO_NUM
EMPLOYEE_NO_NUM
INCOME
SHAP value for
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_BUSINESS_AG
TOTAL_SCORE
SHAP value for BUSINESS_AGE_TO_AGE_RATIO
BUSINESS_AGE_TO_AGE_RATIO
LENGTH_RELATIONSHIP_WITH_CLIENT
SHAP value for
LENGTH_RELATIONSHIP_WITH_CLIENT_TO_WORK_SENIORI
BUSINESS_AGE

Ostateczny wybór

GaussianNB

	precision	recall	f1-score	support
0	0.97	0.82	0.89	2734
1	0.47	0.85	0.60	501
accuracy			0.83	3235
macro avg	0.72	0.84	0.75	3235
weighted avg	0.89	0.83	0.85	3235

ROC AUC score: 0.8361853469359743

Dziękujemy za uwagę