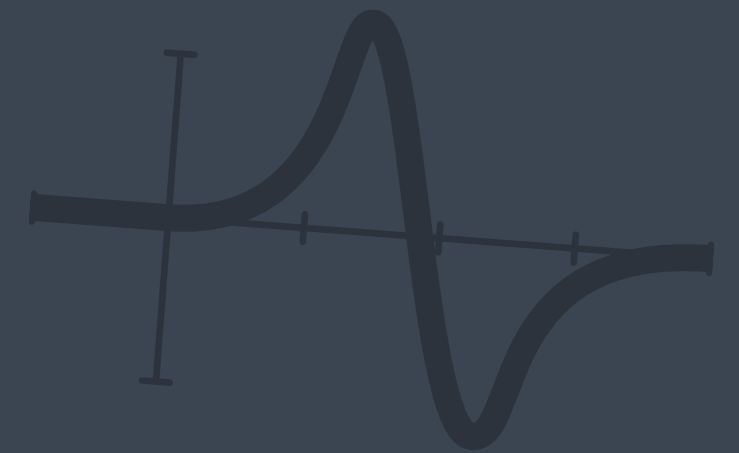


CREDIT CARD FRAUD DETECTION

WOJCIECH ŁUKASZYK, WIKTORIA KAWA





OPIIS PROBLEMU

Budowa modelu wykrywającego oszustwa bankowe

Kolumna 'Class'

```
df['Class'].value_counts()
```

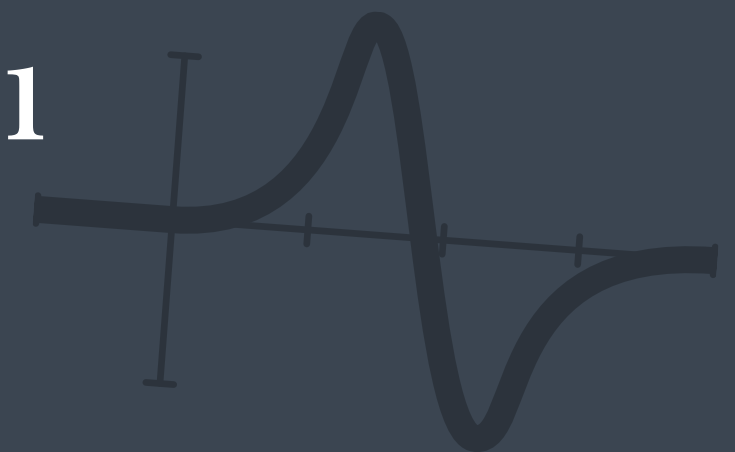
Class

'0' 284315

'1' 492

Name: count, dtype: int64

Czyli budowa modelu
modelu klasyfikacji
binarnej 0-1





PROBLEM BIZNESOWY

BŁĘDNI
ZAKLASYFIKOWANY
PRZYPADEK

=>

STRATA
BANKU

Jeżeli transakcja poprawna zaklasyfikowana zostanie jako oszustwo, to bank traci tyle pieniędzy ile kosztuje zweryfikowanie transakcji.

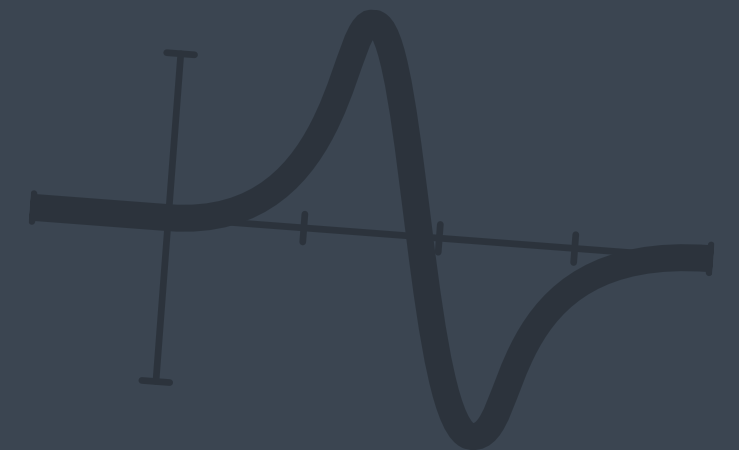
Jeżeli transakcja, która jest oszustwem, zaklasyfikowana będzie jako transakcja poprawna, wówczas bank ponosi straty równe wartości transakcji



PREPROCESSING

Tabela posiada kolumny: Time, Amount, Class, V1, V2, ..., V28

Kolumny V1-V28 są po przekształceniach PCA.





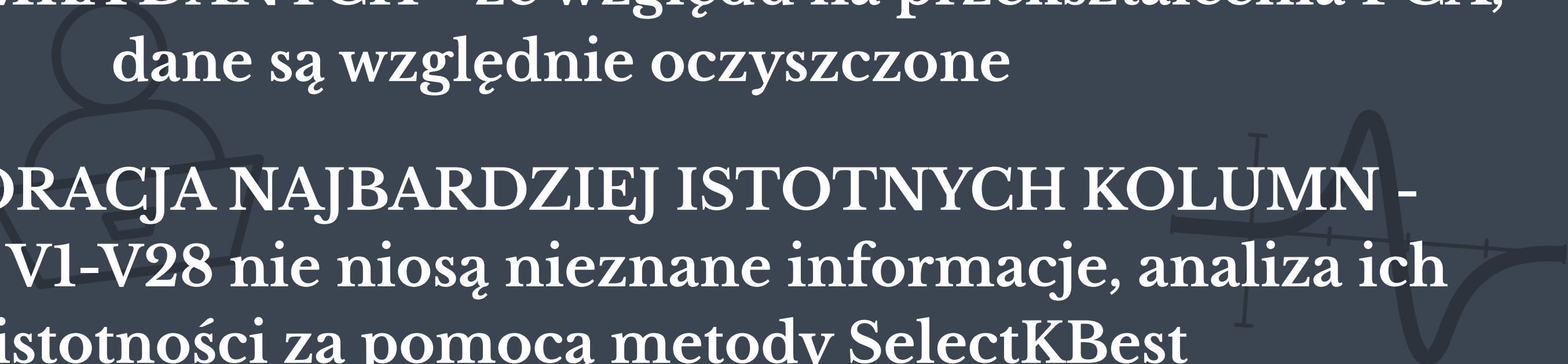
PREPROCESSING

USUNIĘCIE KOLUMNY TIME - nie niesie ze sobą żadnych informacji

MAŁA ILOŚĆ OSZUSTW - tylko 0.172% rekordów z Class=1

CZYSTA RAMKA DANYCH - ze względu na przekształcenia PCA, dane są względnie oczyszczone

EKSPLORACJA NAJBARDZIEJ ISTOTNYCH KOLUMN -
kolumny V1-V28 nie niosą nieznanej informacji, analiza ich
istotności za pomocą metody SelectKBest





PREPROCESSING

MAŁA ILOŚĆ OSZUSTW - nie będziemy stosować żadnych metod balansowania danych, problem niezbalansowanych danych będzie znoszony wprowadzając miary

Ze względu na większą wagę błędów FN niż FP, będziemy stosować miarę F2. Zwiększa ona wagę Recall nad Precission, co jest u nas bardzo ważne.



Oczywiście będziemy sprawdzać też klasycznie ROC AUC.



PRZEDSTAWIENIE MODELI

KLASYFIKATOR GŁOSUJĄCY

Łączy trzy klasyfikatory -
LogisticRegression,
RandomForestClassifier oraz SVC

WYNIKI:

- Recall: 0.596 (niski)
- Precision: 0.973
- F2-score: 0.645
- ROC AUC: 0.969
- Koszt: -10 922.98

Niski Recall => Model nie radzi sobie
dobrze z wykrywaniem oszustw

PRZEDSTAWIENIE MODELI

BAGGING Z DRZEWAMI DECYZYJNYMI

Pipeline z: CustomTransformer
(Funkcja usuwająca kolumnę Time),
SelectKBest, BaggingClassifier

WYNIKI:

- Recall: 0.771
- Precision: 0.914
- F2-score: 0.795
- ROC AUC: 0.927
- Koszt: -11 053.98

Lepszy od wcześniejszego, zwłaszcza
pod kątem recall i F2-score

PRZEDSTAWIENIE MODELI

BAGGING Z SVM

Pipeline z: CustomTransformer
(Funkcja usuwająca kolumnę Time),
SelectKBest, SVM

WYNIKI:

- Recall: 0.708
- Precission: 0.882
- F2-score: 0.737
- ROC AUC: 0.897
- Koszt: -11 016.98

Gorszy od poprzedniego, zwłaszcza pod
względem ROC AUC oraz recall

PRZEDSTAWIENIE MODELI

GRADIENT BOOSTING

Pipeline z: CustomTransformer
(Funkcja usuwająca kolumnę Time),
SelectKBest,
GradientBoostingClassifier

WYNIKI:

- Recall: 0.536
- Precission: 0.789
- F2-score: 0.567
- ROC AUC: 0.699
- Koszt: -10 943.98

Najgorszy z testowanych modeli. Niski
zarówno AUC ROC jak i F2.



PRZEDSTAWIENIE MODELI

Dzięki dostrajaniu hiperparametrów, za pomocą GridSearch udało się znaleźć najlepszy z modeli, którego wyniki widac poniżej:

RANDOM FOREST

Pipeline z: CustomTransformer
(Funkcja usuwająca kolumnę Time),
SelectKBest,
RandomForestClassifier

WYNIKI:

- Recall: 0.797
- Precission: 0.939
- F2-score: 0.821
- ROC AUC: 0.980
- Koszt: -11 053.98

Najlepszy z dotychczasowych modeli



PRZEDSTAWIENIE MODELI

Dzięki dostrajaniu hiperparametrów, za pomocą GridSearch udało się znaleźć najlepszy z modeli, którego wyniki widac poniżej:

EXTRA TREES

Pipeline z: CustomTransformer
(Funkcja usuwająca kolumnę Time),
SelectKBest,
ExtraTreesClassifier

WYNIKI:

- Recall: 0.794
- Precission: 0.946
- F2-score: 0.820
- ROC AUC: 0.950
- Koszt: -11 052.98

Bardzo podobny do RandomForest, dużo szybciej się uczy.

FEATURE IMPORTANCE

EXTRA TREES

V16: 0.1787053367857951
V13: 0.16627298801571558
V11: 0.15434571921621423
V9: 0.10400111260327595
V15: 0.08555446427871775
V10: 0.08002284631079425
V3: 0.06790183196558551
V17: 0.05893224310666333
V2: 0.055897026592978995
V6: 0.04836643112425932

RANDOM FOREST

V16: 0.24813516934736302
V13: 0.23473923948040545
V9: 0.12679898386332325
V11: 0.1262488450517961
V10: 0.06731866486944989
V3: 0.05758845302869832
V15: 0.04628292962579698
V6: 0.04265098601872183
V2: 0.028712945593879595
V17: 0.021523783120565607

Oba te modele używają tych samych kolumn, w lasach możemy wyróżnić dwie najważniejsze cechy V16, V13, w extra-trees możemy wyróżnić 3 (dwie te same i V11)




TESTOWANIE MODELI

Testujemy RandomForest oraz ExtraTrees

RANDOM FOREST

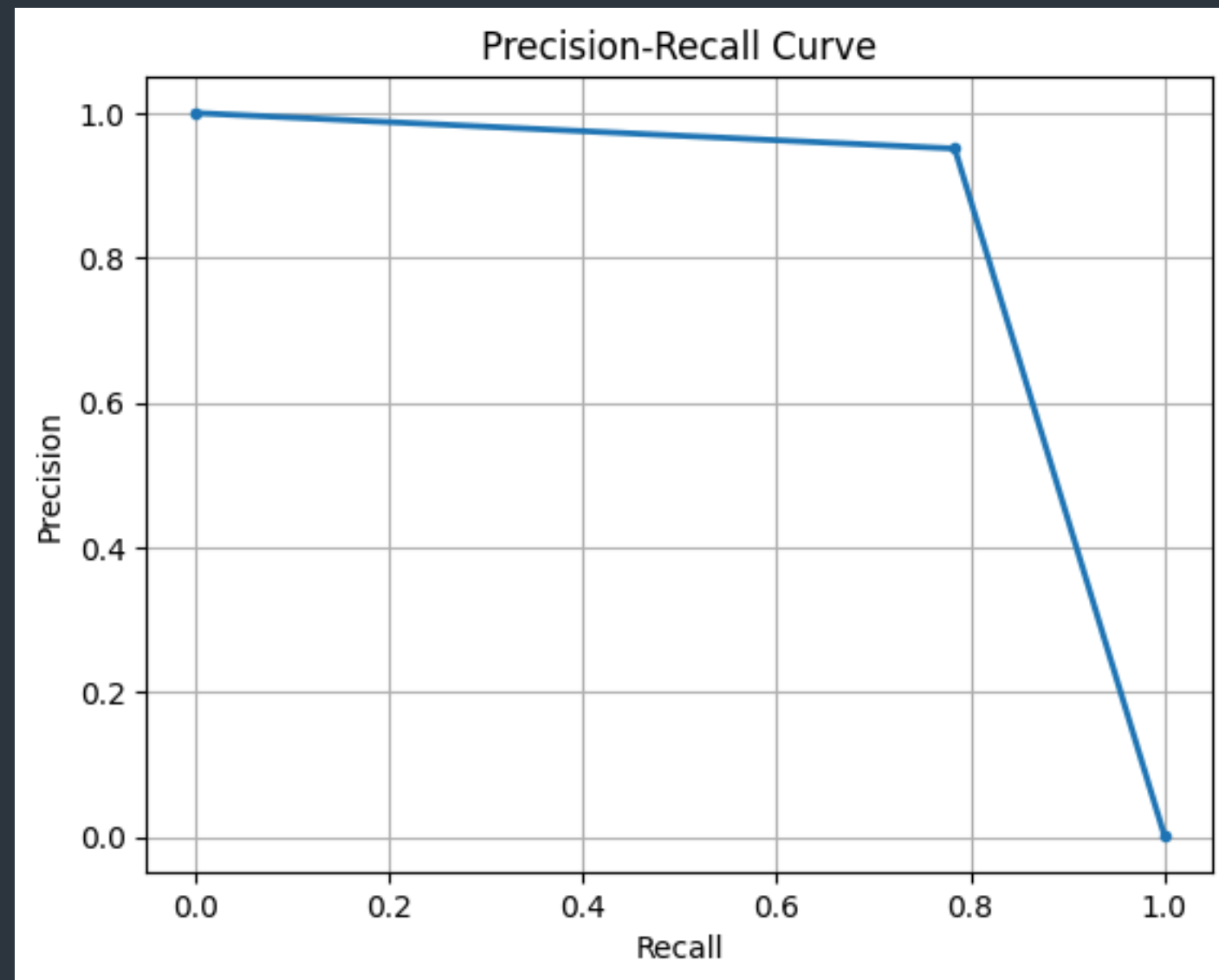
- Recall: 0.784
- F2-score: 0.812
- ROC AUC: 0.891
- Koszt: -7 838.66

EXTRA TREES

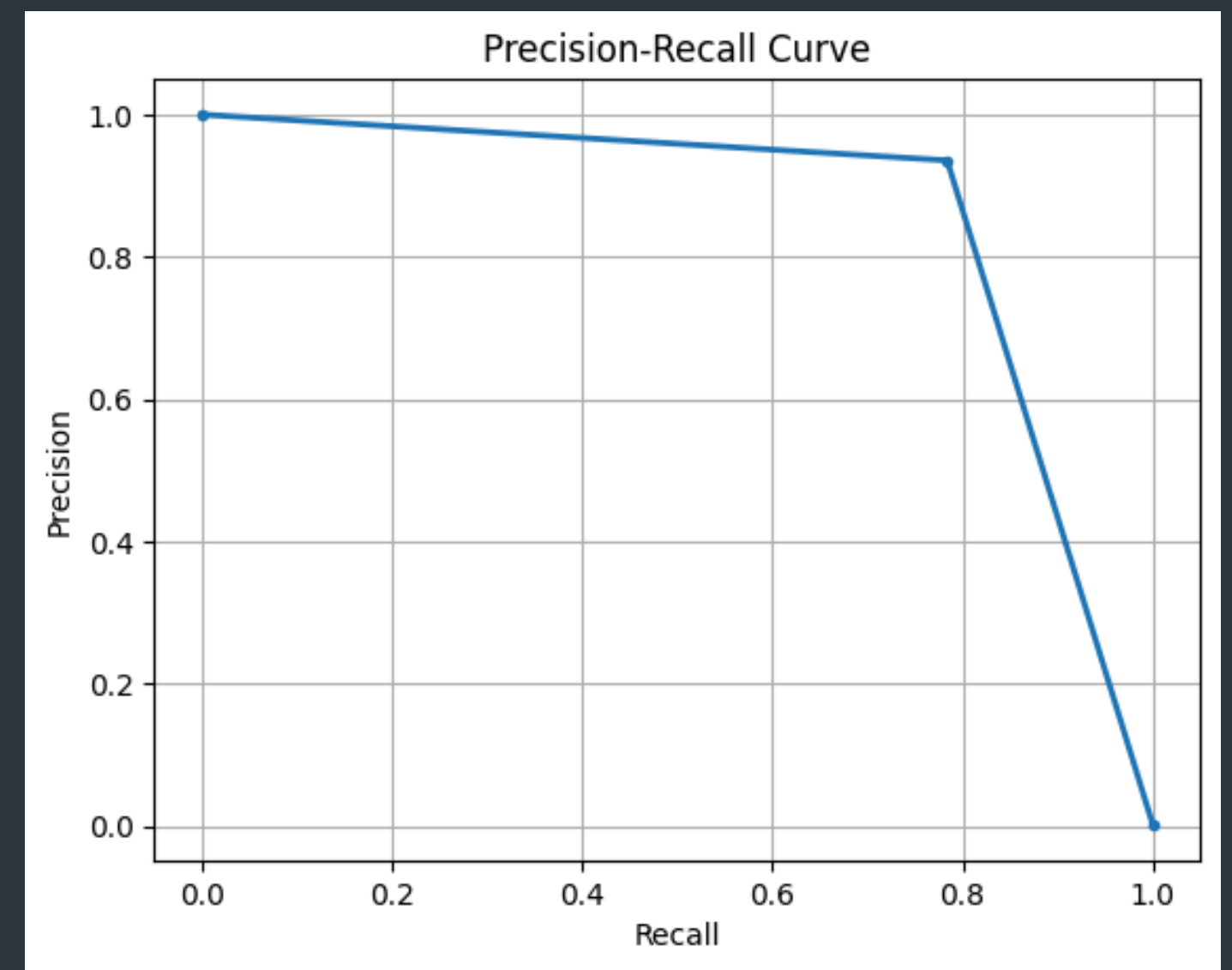
- Recall: 0.784
 - F2-score: 0.810
 - ROC AUC: 0.891
 - Koszt: -7 858.66
- 

TESTOWANIE MODELI

RANDOM FOREST



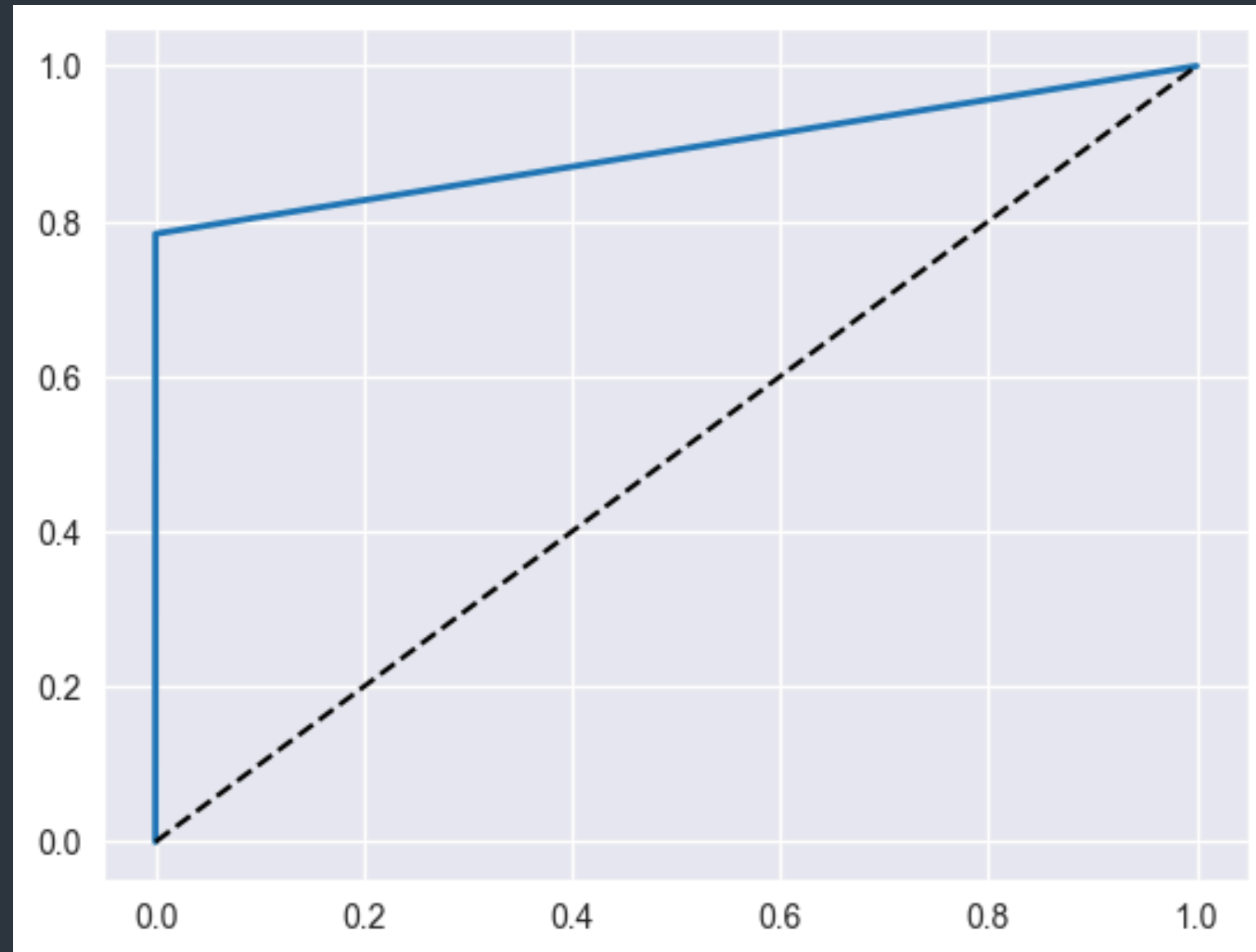
EXTRA TREES



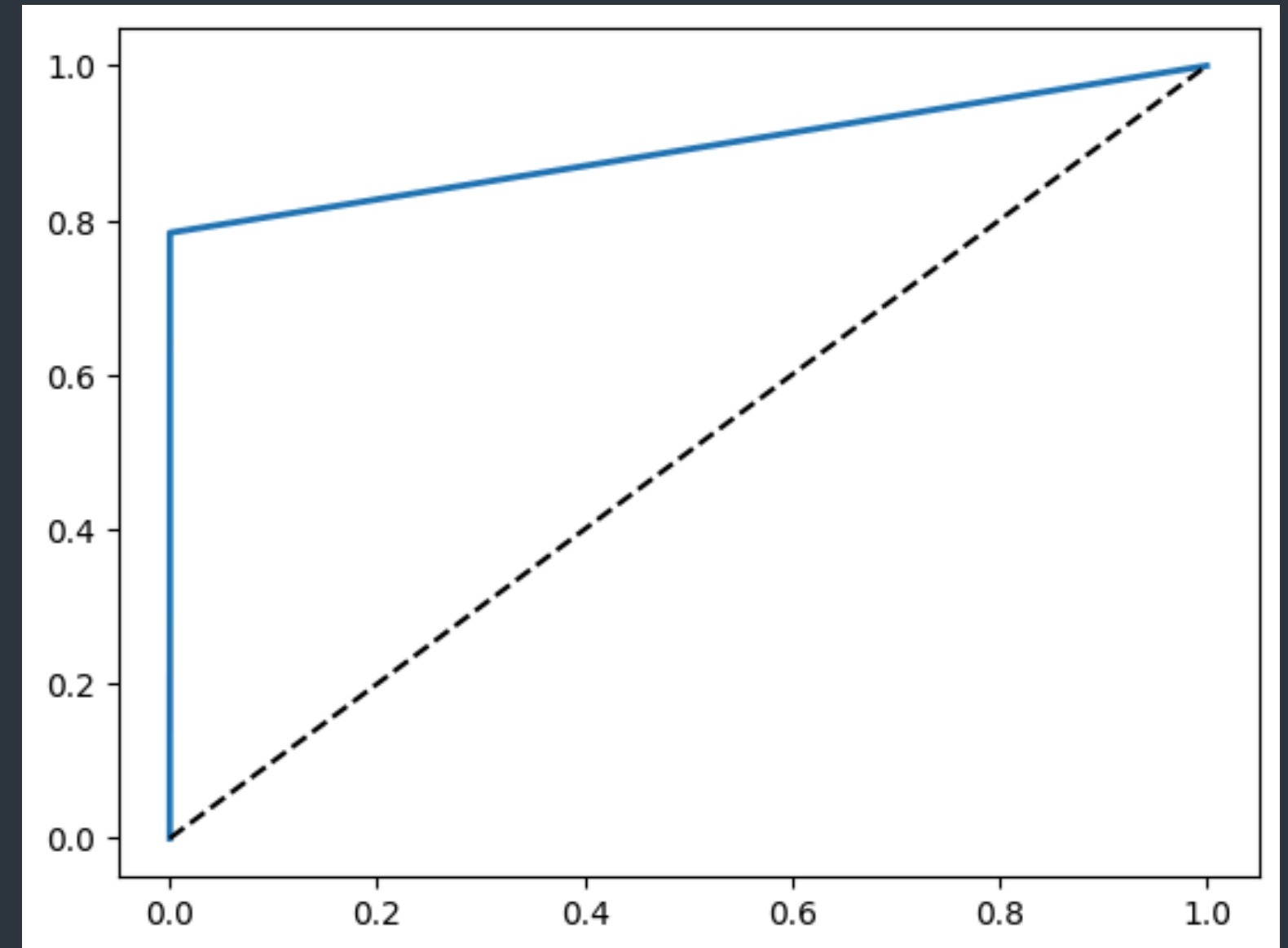
TESTOWANIE MODELI

Krzywa ROC

RANDOM FOREST



EXTRA TREES





TESTOWANIE MODELI


Testujemy RandomForest oraz ExtraTrees

RANDOM FOREST

```
array([[42645,    3],  
       [   16,   58]])
```

EXTRA TREES

```
array([[42644,    4],  
       [   16,   58]])
```



RandomForest posiada o 1 mniej FP i ma o 1 więcej poprawnie zaklasyfikowanych rekordów. Jednak różnica ta jest niewielka, a błąd jaki został popełniony przez ExtraTrees jest FP, czyli model zaklasyfikował jako oszustwo transakcję poprawną. Jest to mniej krzywdzący dla banku błąd. Modele te działają podobnie, ExtraTrees liczą się odczuwalnie szybciej.

INTERPRETACJA PRZY UŻYCIU LIME

