Projekt 1

Wstęp do uczenia maszynowego

Mikołaj Jakubowski, Marceli Korbin, Mariusz Słapek

Politechnika Warszawska

m.jakubowski@student.mini.pw.edu.pl
m.korbin@student.mini.pw.edu.pl
m.slapek@student.mini.pw.edu.pl

20 kwietnia 2020

Plan prezentacji

1. Badania eksploracyjne
2. Preprocessing danych
3. Modelowanie i ewaluacja
4. Wyniki
5. Feature importance

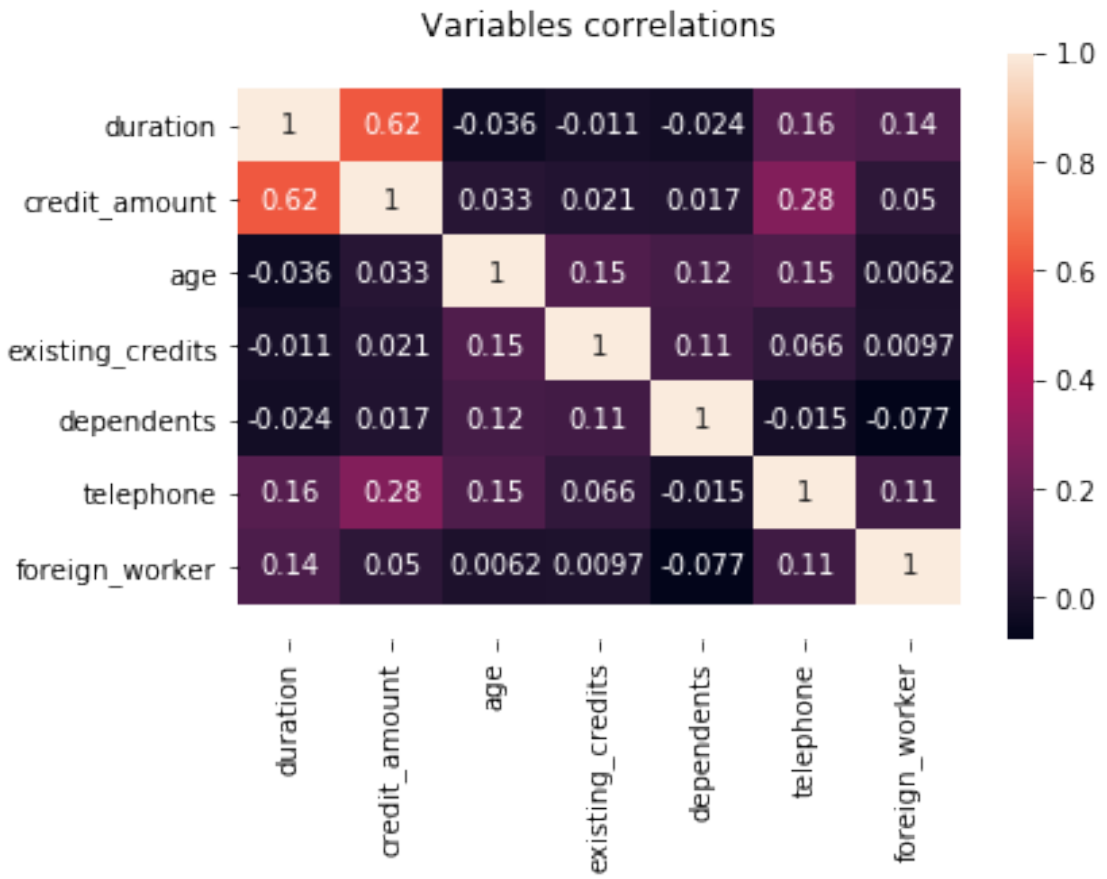
Wybrany zbiór danych

German Credit Data:

1. zawiera tysiąc przykładów czy kredyt został spłacony (70% przypadków pozytywnych)
2. nie zawiera danych brakujących
3. zbiór danych pochodzi z 1994 roku
4. zawiera 21 atrybutów (dużo zmiennych wymaga target encoding)

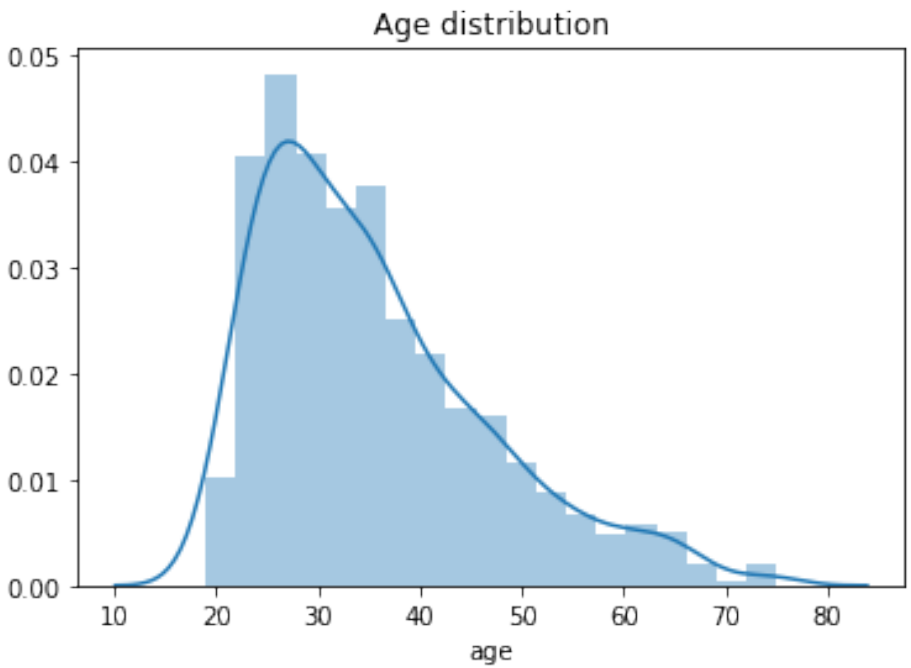
Badania eksploracyjne

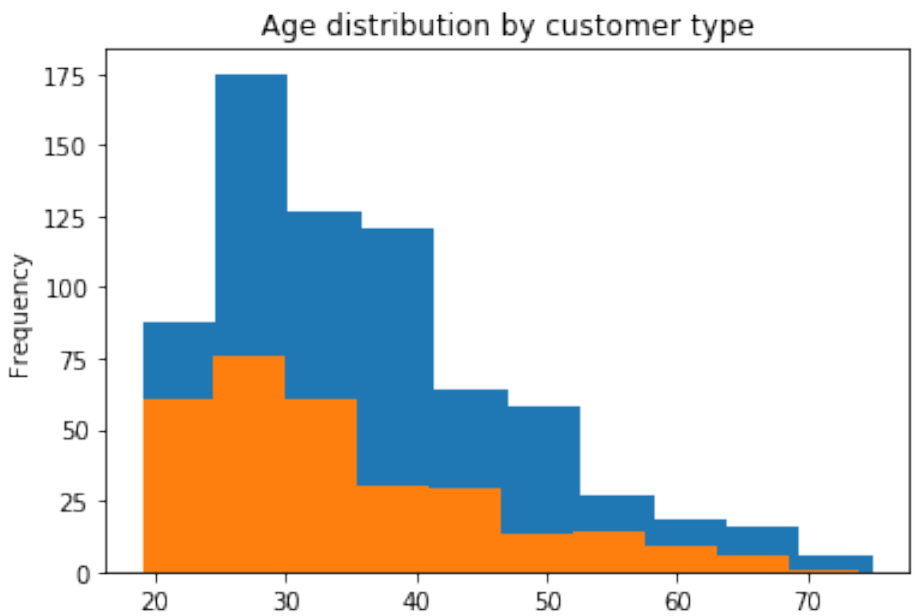
- zmienne credit amount oraz duration silnie skorelowane (0.62)



Badania eksploracyjne

- credit jest w większość brany przez trzydziestolatków
- niezawodność spłacania jest niezależna od wieku

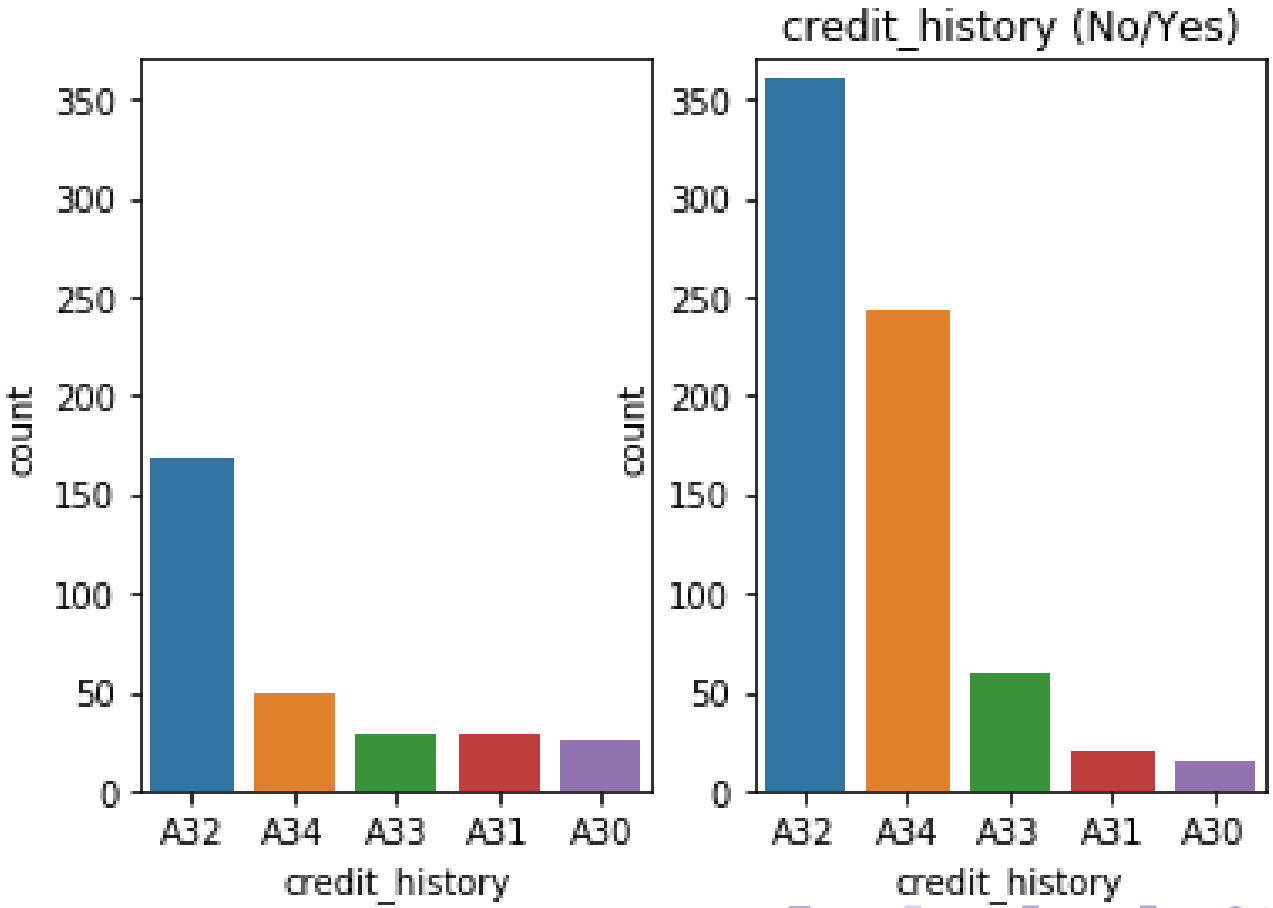




Badania eksploracyjne

- klienci, którzy spłacali kredyty wcześniej przy następnym kredycie mają z tym problem (i vice-versa) - zaskakujący fakt

- A30: no credits taken/ all credits paid back duly
- A31: all credits at this bank paid back duly
- A32: existing credits paid back duly till now
- A33: delay in paying off in the past
- A34 : critical account/ other credits existing (not at this bank)



Preprocessing danych

1. zmapowanie zmiennych binarnych (yes/no) na 1/0
2. brak wyraźnych outlierów (w naszym projekcie nie było to problemem)
3. dodanie płci (z kolumny *personal* gdzie płeć była połączona z stanem cywilnym)
4. normalizacja zmiennych ciągłych
5. zamiana zmiennych kategorycznych przy pomocy *target encodingu* oraz innych metod (bardziej omówione w następnym rozdziale)

Podział zbioru na testowy i treningowy

Dividing into train and test

To divide set so in both parts with have simmilar amounts of big and small credits we need to put them into groups:
(in traditional splits, randomization makes results appear very uneven)

In [3]:
```
from sklearn import preprocessing

x = data[['credit_amount']].values.astype(float)
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
data['amount_groups'] = x_scaled
bins = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8]
data['amount_groups'] = np.digitize(data['amount_groups'], bins)
unique, counts = np.unique(data['amount_groups'], return_counts=True)
dict(zip(unique, counts))
```

Out[3]:
```
{1: 445, 2: 293, 3: 97, 4: 80, 5: 38, 6: 19, 7: 14, 8: 8, 9: 6}
```

In [4]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data.drop(['customer_type', 'amount_groups'], axis=1),
 data.customer_type, test_size=0.20, stratify = data[['amount_groups', 'customer_type']])
```

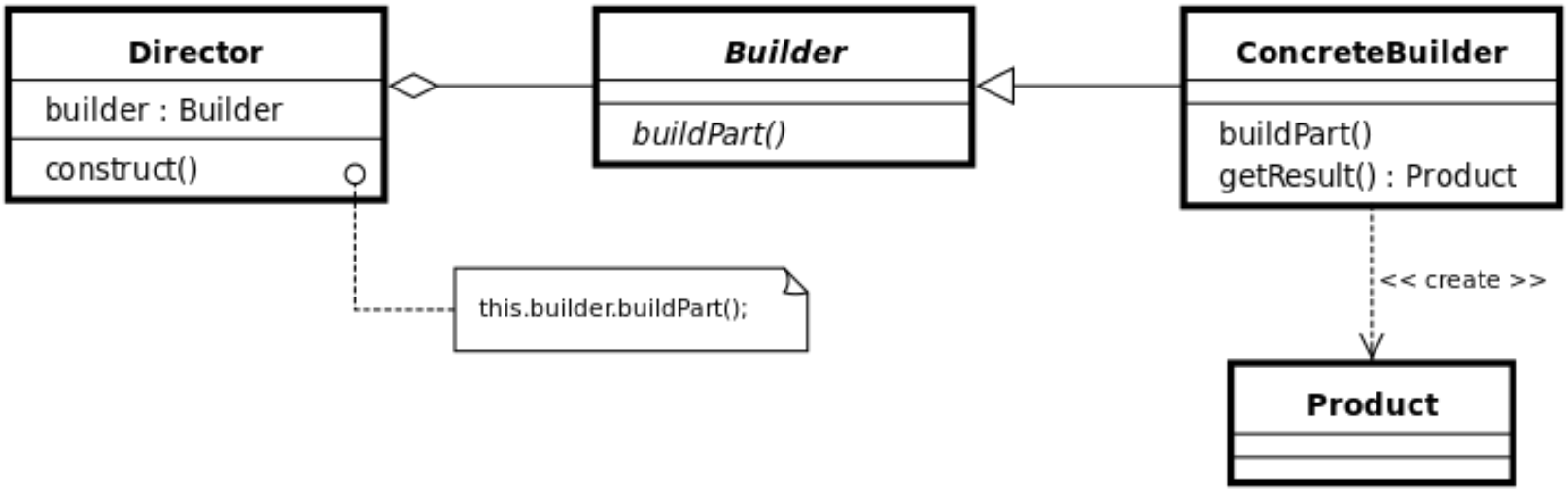
Enkodowanie zmiennych kategorycznych

Stworzenie generycznej funkcji, która dla danych kolumn endodowała zmienne na wiele sposobów, m.in.:

1. one-hot encoding
2. binary
3. ordinal
4. target encoding
5. polynomial
6. ...

Pipeline do tworzenia modeli

W naszym rozwiązaniu w celu łatwej ewaluacji modeli skorzystaliśmy z Buildera - wzorca projektowego, który pozwala w łatwy sposób dodawać wiele modeli a następnie ewaluować je na konkretnych zbiorach danych.



Pipeline do tworzenia modeli - przykład

Create different types of models

In [20]:
```
director = Director()
builder = ConcreteBuilder()
director.builder = builder
```

In [21]:
```
director.add_model('logistic regression', {'penalty': 'l1', 'fit_intercept': 'False'})
director.add_model('gradient boost', {})
director.add_model('xgboost', {})
director.add_model('random forest', {})
```

In [22]:
```
models = director.get_all_models()
```

Wykorzystanie cross-validation

Random Forest tuning

In [46]:
```
parameters = {
    "min_samples_split": [0.01, 0.03, 0.05],
    "min_samples_leaf": [0.01, 0.02, 0.03],
    "max_depth":[3,5,8],
    "max_features":["log2","sqrt"],
    "criterion": ["gini",  "mae"],
    "n_estimators":[10, 100, 150],
    "ccp_alpha":[0.0, 0.01, 0.1]
    }
```

In [47]:
```
df_train, df_test = multiEnc(X_train, X_test, y_train, columns_enc, rforest_enc)
```

In [48]:
```
grid = GridSearchCV(estimator=models[2], param_grid = parameters, scoring = 'f1', cv=4, n_jobs=-1)
grid_result = grid.fit(df_train, y_train)
```

In [49]:
```
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

```
Best: 0.854381 using {'ccp_alpha': 0.0, 'criterion': 'gini', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 0.01,
'min_samples_split': 0.01, 'n_estimators': 150}
```

Główne algorytmy

Podczas testów zobaczyliśmy, iż najlepiej do naszego problemu spisywały się następujące algorytmy:

1. Gradient boosting
2. Random forest
3. XGBoost
4. SVM

Wykorzystywane metryki

1. F1-score
2. metryka biznesowa
3. krzywa ROC (oraz wartość AUC)

Metryka biznesowa

Źródła:

1. According to 'Overseas Business Reports', U.S. Department of Commerce, Breau of International Commerce, 1991 (chapter: 'banking and credit') mortgage interest rate on average was around 10%, for rest type of loans around 13%
2. 'Banking Systems Simulation: Theory, Practice, and Application of Modeling Shocks, Losses, and Contagion', Stefano Zedda (chapter 2.10.1) LGD(loss given default) between 1990 and 2008 was at average 38%

Wyniki

Trafiały nam się modele gdzie F1-score był większy (na poziomie 0.86), ale wówczas metryka biznesowa była gorsza. Z tego powodu szukaliśmy modelu, który dla tych dwóch metryk dawał satysfakcjonujące wyniki. Najlepsze modele (wg tych dwóch metryk):

1. Gradient boosting:
   - F1-score: 0.83217
   - Metryka biznesowa: 0.210441
2. Random Forest:
   - F1-score: 0.83893
   - Metryka biznesowa: 0.137315

Wyniki - przykład

In [20]:
```
f1
```

Out[20]:

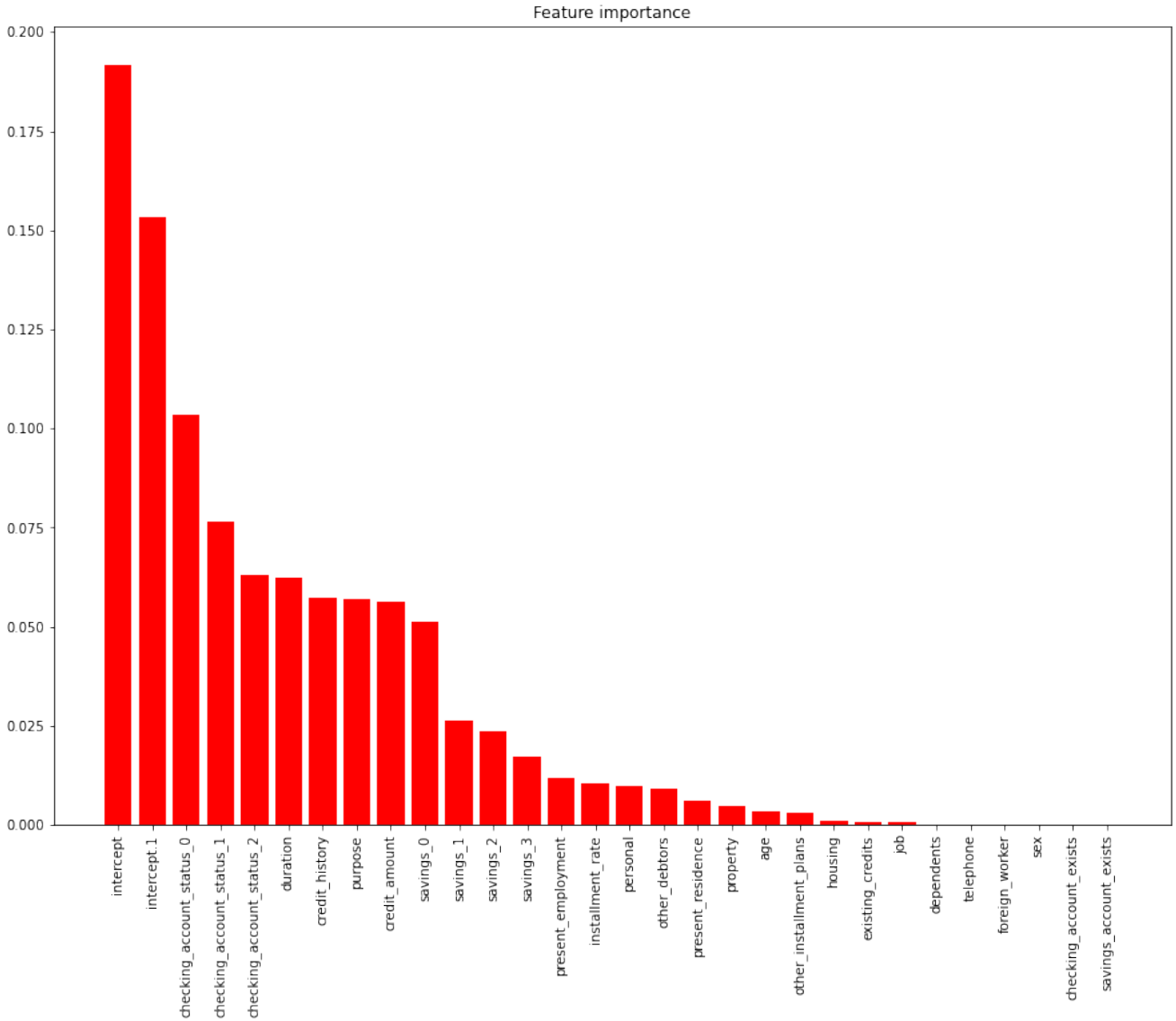
	l	j	m
p	0.833333	0.833333	0.833333
h	0.8223	0.8223	0.8223
d	0.840278	0.840278	0.843206
l	0.827586	0.827586	0.827586
j	0.827586	0.827586	0.827586

In [21]:
```
buss
```

Out[21]:

	l	j	m
p	0.205109	0.205109	0.205109
h	0.12244	0.12244	0.12244
d	0.109999	0.109999	0.188288
l	0.0454018	0.0454018	0.0454018
j	0.0454018	0.0454018	0.0454018

Feature importance - wykres 1



Feature importance - wykres 2