RAPORT WALIDACYJNY
WUM PROJEKT 1 - KLASYFIKACJA

Zespół walidujący (autorzy raportu): Zofia Kamińska, Mateusz Deptuch

Zespół modelujący: Karolina Dunal, Natalia Choszczyk

Dane i temat projektu: Przewidywanie cen telefonów (price range 0-3) na podstawie danych fizycznych i technicznych urządzenia

1. KM1

1.1. EDA

Feedback:

- podział danych na zbiory treningowy+walidacyjny/testowy **poprawny**

- sprawdzenie braku danych **uwaga:** poza nullami warto również sprawdzić czy są zerowe wartości w kolumnach takich jak width itd.

```
rows_with_zero_height = mobile_df[mobile_df['sc_w'] == 0]

# Get the count of rows with height equal to 0
count_zero_height = len(rows_with_zero_height)

print(f"Number of rows with width equal to 0: {count_zero_height}")
```
✓ 0.0s

```
Number of rows with width equal to 0: 124
```

- ciekawy pomysł obliczenia screen area na podstawie wysokości i szerokości
uwaga: brakujące wartości (zera) mają spory wpływ na analizowane dane

statystyki dla screen area przed usunięciem zer:

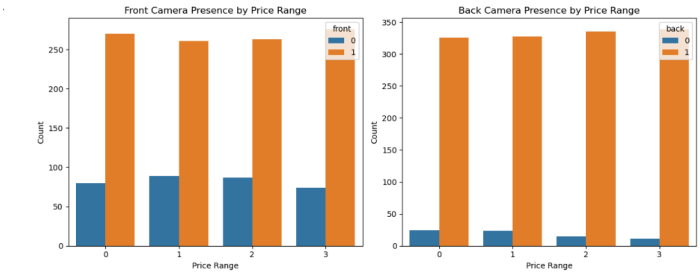
```
count    1400.000000
mean       80.690000
std        77.320899
min         0.000000
25%        18.750000
50%        55.000000
75%       126.000000
max       342.000000
dtype: float64
```

po usunięciu zer:

```
Screen Area Description:
count    1276.000000
mean       88.531348
std        76.584674
min         5.000000
25%        28.000000
50%        63.500000
75%       130.000000
max       342.000000
Name: screen_area, dtype: float64
```

sugestia: przy tak dużej ilości brakujących wartości screen width może warto policzyć rozdzielczość ekranu bazując na wartościach pikselowych i brać to pod uwagę zamiast powierzchni ekranu

- zmienne front camera i back camera, **sugestia:** może warto spróbować osobno przeanalizować telefony które posiadają lub nie aparat (rozdzielić wartości gdzie wartości w mpix jest 0),



uwaga: 330 nie ma kamery przedniej, 73 nie mają kamery tylnej, możliwe że po tym łatwo byłoby wyłapać najtańsze telefony - **do sprawdzenia**

- box ploty **sugestia:** dla continuous values, skoro boxploty dla każdego price range'a wyglądają podobnie (czasami identycznie) może warto zamiast tego skorzystać z violin plota **sugestia 2:** może łatwiej byłoby wyłapywać ciekawe różnice gdyby na każdym plocie była jedna cecha dla różnych price range'ów a nie różne cechy dla jednego price range'a.

- macierz korelacji **uwaga:** może warto dodać liczby dla lepszej czytelności

sugestia: zależności między zmiennymi ciągłymi dobrze by było zwizualizować na scatter plotach
uwaga: niektóre wykresy powinny być dokładniej opisane, wyciągnięte jakieś wnioski

*Większość naszych sugestii zostało wzięte pod uwagę do KM1 co wpłynęło na dokładniejszą analizę danych wejściowych. Nie dodano tylko liczb na macierzy korelacji jako, że modelarkom zależało by pokazać, że korelacje są po prostu małe a nie skupiać się na dokładnych wartościach. Nie skupiono się bardziej na kamerach, gdyż nie było widać silnej korelacji między ceną telefonu a posiadaniem kamery.

2. KM2

2.1. FEATURE ENGINEERING

Feedback:

- użycie Random Forest do sprawdzenia użyteczności kolumn (feature importance), **dobry pomysł**
- **dobrze,** że zostały sprawdzone różne warianty, z różną liczbą usuniętych kolumn. Sprawdzone warianty kolumn:

1) all columns
2) all columns, standardized
3) only features with importance greater than 0.05 (“>0.05”)
4) only features with importance greater than 0.032 (“>0.032”)
5) only features with importance greater than 0.02 (“>0.02”)
6) only RAM & Battery Power

ram	0.411480
battery_power	0.072808
px_height	0.070469
px_width	0.057439
mobile_wt	0.047672
int_memory	0.038276
clock_speed	0.036336
talk_time	0.033699
pc	0.033030
sc_w	0.032927
fc	0.032733
sc_h	0.031672
m_dep	0.030613
n_cores	0.025984
touch_screen	0.008038
blue	0.007855
dual_sim	0.007816
four_g	0.007589
wifi	0.007120
three_g	0.006444

feature importance using Random Forest

- standaryzacja, użyto Standard Scaler dla pełnego zestawu danych, **uwaga:** warto przetestować inne metody standaryzacji takie jak box cox itp. i również spróbować standaryzacji na wariantach z usuniętymi kolumnami

2.2. FIRST MODELS

Skupiliśmy się za weryfikacji tego czy na zbiorze walidacyjnym wychodzą podobne wyniki jak na zbiorze modelarzy, by wyłapać ewentualne zjawisko overfittingu.

Feedback:

2.2.1. Dummy Classifier

Modelarze osiągnęli performance 0.25 accuracy oraz f1 score. Na zbiorze walidacyjnym wynik jest nieco wyższy, 0.32, ale wciąż jest to bardzo niski wynik, czego można było się spodziewać.

2.2.2. Logistic Regression

We wnioskach napisano, że najlepiej działający model jest dla wariantu “all features”, lecz z wyników wynika że w rzeczywistości był to wariant “>0.05”.
Na zbiorze walidatorskim najlepiej sprawdził się ustandaryzowany wariant “all features” a także “RAM & battery power”, lecz nadal wartości accuracy i f1 score wahają się w granicach 30%.

2.2.3. Decision Tree

Na zbiorze modelarskim top 2 rezultaty uzyskane dla “>0.05” i “0,032”, ten pierwszy z wynikiem 0.86 (accuracy, f1 score), **UWAGA:** na zbiorze walidatorskim ten sam wariant uzyskał 0.79, spora różnica więc trzeba uważać. Top dwa wyniki na zbiorze walidatorskim:

Result for features with importance greater than 0.032:

	precision	recall	f1-score	support
0	0.93	0.90	0.92	31
1	0.84	0.81	0.83	32
2	0.68	0.68	0.68	31
3	0.76	0.81	0.79	32
accuracy			0.80	126
macro avg	0.80	0.80	0.80	126
weighted avg	0.80	0.80	0.80	126

Result for all columns, standaraised:

	precision	recall	f1-score	support
0	0.97	0.90	0.93	31
1	0.80	0.75	0.77	32
2	0.64	0.74	0.69	31
3	0.81	0.78	0.79	32
accuracy			0.79	126
macro avg	0.80	0.79	0.80	126
weighted avg	0.80	0.79	0.80	126

2.2.4. SVM

Najlepsze rezultaty na zbiorze modelarzy otrzymano dla “>0.05” i “0.32” z rezultatami 0.96 (accuracy i f1 score) i 0.95 odpowiednio. Na zbiorze walidatorskim dla tych wariantów uzyskano podobne wyniki. **UWAGA:** Najlepszy wynik w naszym przypadku otrzymaliśmy dla wariantów “all columns” i “>0.02”, więc dla większej ilości kolumn. Jest to najlepszy dotychczas wynik 0.97 (accuracy i f1 score).

2.2.5 Random Forest

Najlepiej performujące warianty kolumn:

Result for features with importance greater than 0.05:

	precision	recall	f1-score	support
0	0.94	0.97	0.95	31
1	0.88	0.91	0.89	32
2	0.87	0.87	0.87	31
3	0.97	0.91	0.94	32
accuracy			0.91	126
macro avg	0.91	0.91	0.91	126
weighted avg	0.91	0.91	0.91	126

Result just for RAM and battery power:

	precision	recall	f1-score	support
0	0.94	0.97	0.95	31
1	0.88	0.91	0.89	32
2	0.90	0.84	0.87	31
3	0.94	0.94	0.94	32
accuracy			0.91	126
macro avg	0.91	0.91	0.91	126
weighted avg	0.91	0.91	0.91	126

sugestia: przy analizie wyników może warto zwrócić uwagę na to która klasa jest odgadywana najlepiej

Poniżej przedstawione są również dokładne wartości dla każdego z modeli:

(najlepsze wyniki każdego z modeli zostały pogrubione)

model/version	accuracy (modelarze)	accuracy (walidatorzy)	f1 score (modelarze)	f1 score (walidatorzy)
Logistic Regression				
all columns	0.27	0.24	0.26	0.24
all columns, standardized	0.22	0.25	0.22	0.25
* >0.02	**0.28**	**0.31**	**0.28**	**0.31**
* >0.032	0.24	0.23	0.24	0.23
* >0.05	0.27	0.26	0.27	0.26
RAM & Battery Power	**0.28**	0.21	0.27	0.21
Decision Tree				
all columns	0.82	0.79	0.82	0.79
all columns, standardized	0.82	**0.81**	0.82	**0.81**
* >0.02	0.80	0.74	0.80	0.74
* >0.032	**0.86**	0.80	**0.86**	0.80
* >0.05	0.85	0.80	0.85	0.80
RAM & Battery Power	0.75	0.77	0.75	0.77
SVM				
all columns	0.94	**0.97**	0.94	**0.97**
all columns, standaraised	0.84	0.75	0.84	0.76
* >0.02	0.94	**0.97**	0.94	**0.97**
* >0.032	0.95	0.96	0.95	0.96
* >0.05	**0.96**	0.95	**0.96**	0.95
RAM & Battery Power	0.81	0.79	0.81	0.79

Random Forest Classifier				
all columns	0.84	0.83	0.84	0.83
all columns, standaraised	0.84	0.85	0.83	0.85
* >0.02	0.84	0.90	0.84	0.90
* >0.032	**0.90**	**0.91**	**0.90**	**0.91**
* >0.05	0.89	**0.91**	0.89	**0.91**
RAM & Battery Power	**0.90**	**0.91**	**0.90**	**0.91**

zbiór modelarzy
zbiór walidacyjny
*features with importance greater than

Ogółem osiągane wyniki modeli na zbiorze modelarzy i zbiorze walidatorskim są podobne. W przypadku niektórych modeli (np.SVM) inne (niż na zbiorze modelarzy) warianty usunięcia kolumn dają najlepsze wyniki, ale są to różnice mieszczące się w zakresie 1% accuracy/f1 score. Generalnie mimo różnic SVM pozostaje najlepiej performującym modelem, **sugestia:** może warto poszukać więcej bardziej złożonych modeli, lub takich których, działanie będzie bardziej przystosowane konkretnie do tych danych (jedna kolumna "RAM" dominująca w sensie korelacji, reszta mniej znacząca)

*Większość naszych sugestii została wzięta pod uwagę do KM2. Z powodu problemów technicznych nie udało się tylko modelarzom pokazać innej metody standaryzacji (było to robione, nie dawało lepszych rezultatów, ale plik niestety przepadł by móc pokazać). Reszta sugestii została wzięta pod uwagę.

3. KM3

3.1. ADVANCED MODELING

Zespół modelarzy testował bardziej złożone modele (ensemble methods) dla różnych kombinacji estymatorów.

1) DecisionTree, RandomForest, SVC
2) RandomForest, KNN, SVC
3) DecisionTree, RandomForest, SVC

Nadal testowane są dwa warianty, wszystkie kolumny oraz ‘only features with importance greater than 0.02 (“>0.02”)’

3.1.1 Voting

Zarówno dla soft votingu, hard votingu, jak i wariantu z wagami “weights” wyniki na zbiorze walidatorskim były podobne, bądź nieco lepsze (różnica maksymalnie 2-3% accuracy). Najlepszy wynik na zbiorze modelarskim otrzymano dla 2) zestawu estymatorów przy wariancie weights z wagami 0.1, 0.1, 0.8 odpowiednio, na zbiorze walidatorskim był to ten sam zestaw, ale na wariancie soft.
Poniżej otrzymane najlepsze wyniki,

modelarzy:

```
accuracy for whole dataset:  0.9421768707482994
accuracy for dataset without chosen columns:  0.9319727891156463
```

walidatorów:

```
accuracy for whole dataset:  0.9523809523809523
accuracy for dataset without chosen columns:  0.9523809523809523
```

- **ciekawa obserwacja:** przy wielokrotnym odpaleniu modelu hard voting, dla 1) estymatorów, za każdym razem otrzymano mniej lub bardziej różne wyniki. Accuracy na pełnym zbiorze wahało się między 91,2-94,4 %, a na ograniczonym “>0.02” w zakresie 88,8-93,6%.

3.1.2 Stacking

Podobnie jak na zbiorze modelarzy, najlepszy wynik otrzymano dla zestawu 3), u modelarzy wynik 96.3%, u nas aż 96.8%, był to wariant:

```
clf = StackingClassifier(estimators=estimators2, final_estimator=RandomForestClassifier())
clf.fit(X_train, y_train).score(X_val, y_val)
```

✓ 1.6s

- **uwaga:** jak poniżej będzie robiona kroswalidacja, to może warto ją zastosować też dla "ensemble methods" zwłaszcza z uwagi na ciekawą obserwację powyżej.

Znaleźliśmy także literówkę i małe braki w kodzie (które najprawdopodobniej wynikły ze zmęczenia modelarzy podczas wrzucania rzeczy do finalnego pliku). Zostawiliśmy też kilka sugestii by dokładniej opisywać niektóre wyniki.

3.1.3. Bagging

Testowane dla DecisionTree oraz RandomForest, wyniki podobne jak na zbiorze modelarzy, nie wykryliśmy żadnego overfittingu, choć generalnie performance raczej kiepski w porównaniu do poprzednich najlepszych (accuracy w granicach 82-89%).

- użyto wizualizacji confusion matrix a także ROC curve do analizy wyników, **dobrze**

3.1.4. Boosting

Wyniki na zbiorze walidacyjnym: lepsze niż u modelarzy, gorsze niż u modelarzy

model/%	**accuracy**	**precision** (weigthed avg)	**recall** (weigthed avg)	**f1-score** (weigthed avg)
AdaBoost	46	57	47	42
AdaBoost >0.02	44	40	43	38
XGBoost	89	89	89	89
XGBoost >0.02	91	91	91	91
CatBoost	87	88	87	87
CatBoost >0.02	**95**	**95**	**95**	**95**

(CatBoost był robiony u modelarzy, ale zakomentowany, więc nie wiemy jakie mieli wyniki)

3.2. PARAMETER TUNING

Zastosowano metody gridsearchcv jak i bayes optimisation dla modeli SVM, RandomForest, DecisionTree i XGBoost, **fajnie** że wykorzystano aż dwie metody, zwłaszcza że sam Bayes zwraca podejrzanie niskie wyniki (niektóre blisko 80%).

3.3. AutoML - tpot

Z ciekawości zastosowaliśmy tpot na danych walidacyjnych i otrzymaliśmy zupełnie inny wynikowy pipeline niż modelarze. Nasz wynik: (u modelarzy logistic regression)

```
Generation 1 - Current best internal CV score: 0.8547619047619047

Generation 2 - Current best internal CV score: 0.880952380952381

Generation 3 - Current best internal CV score: 0.880952380952381

Generation 4 - Current best internal CV score: 0.880952380952381

Generation 5 - Current best internal CV score: 0.880952380952381

Best pipeline: ExtraTreesClassifier(input_matrix, bootstrap=False, criterion=entropy, max_features=0.7000000000000001,
0.9206349206349206
```

3.4. BEST MODELS

Najlepsze z dotychczasowych modeli przetestowano na zbiorach walidacyjnych:

	accuracy modelarze %	accuracy walidacja %
SVM	98	98
Stacking	98	96
Logistic Regression	96	94

a także na testowych:

- modelarze

Model	Accuracy
SVM Model	0.937
Stacking Model	0.937
Logistic Regression Model	0.960
SVM Model - columns removed	0.929
Stacking Model - columns removed	0.944

- walidatorzy

Model	Accuracy
SVM Model	0.968
Stacking Model	0.960
Logistic Regression Model	0.937
SVM Model - columns removed	0.968
Stacking Model - columns removed	0.960

- **ciekawe**, że u nas jest ogólnie lepiej i tylko regresja logistyczna wypada gorzej

3.5. WIZUALIZACJE

Wizualizacje zostały stworzone przy użyciu pakietu lime, lecz wstępnie tylko dla jednej konkretnej obserwacji. Zasugerowaliśmy by porównać kilka, np. biorąc pod uwagę przykłady obserwacji z różnych price range'ów, oraz by podczas analizy skupić się na cechach które wpływają na wyniki w każdym przypadku.

- partial dependence plot, **uwaga:** fajnie że są, ale same w sobie nie są jakoś super efektowne, więc trzeba uważać żeby mieć coś ciekawego o nich do powiedzenia

*Wszystkie uwagi techniczne zostały rozpatrzone i wyniki na zbiorze walidatorskim wzięte pod uwagę przy finalnej analizie. Zrezygnowano z użycia partial dependence plot z uwagi na problemy techniczne z nimi związane a także małą ilość wartości merytorycznych dla tego konkretnego przypadku.