Przewidywanie cen domów

Warsztaty z technik uczenia maszynowego

Piotr Duperas, Aleksandra Mach, Szymon Pawłowski, Jędrzej Wicha

Spis treści

1 Opis projektu

Temat projektu został znaleziony na platformie *kaggle.com*,
link: https://www.kaggle.com/c/house-prices-advanced-regression-techniques.

Zadanie polega na dość dokładnej analizie zależności cen domów w mieście Ames w stanie Iowa od ich różnych cech – zaczynając od typowych jak powierzchnia domu, liczba pokoi, okolica poprzez mniej oczywiste jak np. powierzchnie poszczególnych pomieszczeń, materiały wykorzystywane do budowy fundamentów i ich jakość, stopień wykończenia, a kończąc na dość nieintuicyjnych cechach takich jak liczba kominków, rodzaj dachu, rok wybudowania garażu.

2 Eksploracja danych

Początkowym etapem pracy było dokładne zapoznanie się z danymi oraz ich przygotowanie do dalszej pracy, czyli do budowy modeli. Trzeba było zrozumieć każdą z 79 kolumn, czyli predyktorów, podzielić je na kategoryczne, ciągłe, jakościowe, zmienić zmienne kategoryczne na liczbowe, zestandaryzować odpowiednie zmienne, zweryfikować, czy mamy do czynienia z outlierami oraz uzupełnić braki danych. Etap pracy z samymi danymi zamknęliśmy w jednej funkcji o nazwie *preprocess*.

2.1 Zmienne kategoryczne

Kolumny, w których wyróżniamy po kilka (ewentualnie kilkanaście) różnych kategorii/grup obserwacji.

- **MSSubClass** – typ budynku, uwzględniający ilość pięter, wiek itp.
  Ma 16 kategorii, które zostały zamienione na wartości liczbowe.
- **MSZoning** – typ przestrzeni w jakiej znajduje się budynek, np. przestrzeń przemysłowa, rolnicza, osiedlowa itp.
  Ma 8 kategorii, które zostały zamienione na wartości liczbowe.
- **Street** – typ ulicy.
  Ma tylko dwie kategorie: Gravel i Paved, które zostały zamienione na wartości liczbowe 0, 1.
- **Alley** – typ alei.
  Ma trzy kategorie: Gravel, Paved, No alley access, które zostały zamienione na wartości liczbowe 0, 1, 2.
- **LotShape** – kształt posesji.
  Ma 4 kategorie, które zostały zamienione na wartości liczbowe.
- **LandContour** – typ przekroju posesji, np. płaska, ze spadkiem, z wgłębieniem.
  Ma 4 kategorie, które zostały zamienione na wartości liczbowe.
- **LotConfig** – typ umieszczenia posesji względem ulic i innych posesji, np. na rogu ulicy, sąsiadująca z 3 innymi posesjami itp.
  Ma 5 kategorii, które zostały zamienione na wartości liczbowe.

- **Neighborhood** – nazwa osiedla.
  Ma aż 25 kategorii, które zostały zamienione na wartości liczbowe.

- **Condition1, Condition2** – dodatkowe cechy, np. bliskość dużych ulic, parków itp.
  Mają 9 kategorii, które zostały zamienione na wartości liczbowe 0,1.

- **BldgType** – typ budynku.
  Ma 5 kategorii, które zostały zamienione na wartości liczbowe.

- **HouseStyle** – styl budynku odnoszący się głównie do ilości pięter.
  Ma 8 kategorii, które zostały zamienione na wartości liczbowe.

- **RoofStyle** – rodzaj dachu.
  Ma 6 kategorii, które zostały zamienione na wartości liczbowe.

- **RoofMatl** – rodzaj materiału użytego przy budowie dachu.
  Ma 8 kategorii, które zostały zamienione na wartości liczbowe.

- **Exterior1st** – zewnętrzne wykończenie domu. Ta kolumna zawiera tylko pojedyncze materiały.
  Ma 17 kategorii, które zostały zamienione na wartości liczbowe.

- **Exterior2nd** – zewnętrzne wykończenie domu. Ta kolumna zawiera więcej niż jeden materiał.
  Ma 17 kategorii, które zostały zamienione na wartości liczbowe.

- **MasVnrType** – rodzaj okleiny murarskiej.
  Ma 5 kategorii, które zostały zamienione na wartości liczbowe.

- **Foundation** – rodzaj fundamentu.
  Ma 6 kategorii, które zostały zamienione na wartości liczbowe.

- **Heating** – rodzaj ogrzewania.
  Ma 6 kategorii, które zostały zamienione na wartości liczbowe.

- **CentralAir** – obecność klimatyzacji.
  Ma tylko dwie kategorie: Yes, No, które zostały zamienione na wartości liczbowe 0,1.

- **Electrical** – rodzaj układu elektrycznego.
  Ma 5 kategorii: SBrkr, FuseA, FuseF, FuseP, Mix, które zostały zamienione na wartości liczbowe 0,1,2,3,4.

- **YrSold** – rok sprzedaży domu, ma 5 unikatowych wartości, zamieniamy na wartości liczbowe od 0 do 4.

- **GarageType** – typ garażu, tzn. jego lokalizacja w domu.
  Ma braki danych, które z opisu danych oznaczają brak garażu – nadajemy brakom kategorię 0, pozostałe są pogrupowane w kategorie (od 1 do 6): Detchd (odłączony od domu), CarPort (wiata samochodowa), BuiltIn (wbudowany – część domu), Basment (garaż w piwnicy), Attchd (przyłączony do domu), 2Types (2 lub więcej typów garażu).

- **PavedDrive** – czy posiada podjazd,
  Ma 3 kategorie, tak, nie, częściowy, które zostały zamienione na wartości liczbowe.

- **MiscFeature** – Czy posiada udogodnienia, wśród nich są winda, drugi garaż, szopa, boisko do tennisa. Brak danych oznacza brak zanotowanych udogodnień

- **SaleType** Rodzaj wystawionej oferty sprzedaży – .
  Pogrupowane w kategorie (od 1 do 10): WD (Warranty Deed - Conventional), CWD (Warranty Deed - Cash), VWD (Warranty Deed - VA Loan), New (Home just constructed and sold), COD (Court Officer Deed/Estate), Con (Contract 15percent Down payment regular terms), ConLw (Contract Low Down payment and low interest), ConLI Contract Low Interest), ConLD (Contract Low Down), Oth (Other),

- **SaleCondition** jakość wystawionej oferty sprzedaży – .
  Pogrupowane w kategorie (od 1 do 6): Normal (Normal Sale), Abnorml (Abnormal Sale - trade, foreclosure, short sale), AdjLand (Adjoining Land Purchase), Alloca (Allocation - two linked properties with separate deeds, typically condo with a garage unit), Family (Sale between family members), Partial (Home was not completed when last assessed (associated with New Homes)),

2.2 Zmienne porządkowe/jakościowe

Kolumny, w których także mamy po kilka różnych kategorii/grup obserwacji, ale grupy te mają konkretny porządek (np. od najgorszej jakości do najlepszej).

- **Utilities** – dostępne media.
  Ma 4 kategorie porządkowe (od jedynie elektryczności, po wszystkie udogodnienia, czyli elektryczność, gaz, wodę, kanalizację – każdy kolejny poziom to kolejne udogodnienie dostępne), którym nadajemy wartości liczbowe od 0 do 3.

- **LandSlope** – pochylenie posesji.
  Ma 3 kategorie porządkowe (od drobnego pochylenia, do mocnego), którym nadajemy wartości liczbowe od 0 do 2.

- **OverallQual** – jakość materiałów i wykończenia.
  Ma 10 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 1 do 10.

- **OverallCond** – stan domu.
  Ma 10 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 1 do 10.

- **ExterQual** – jakość (w momencie zakończenia budowy) zewnętrznego wykończenia domu.
  Ma 5 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 4.

- **Exterior2nd** – aktualny stan zewnętrznego wykończenia domu.
  Ma 5 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 4.

- **BsmtQual** – jakość (a konkretnie wysokość) piwnicy. Im wyższa piwnica tym lepsza jakość.
  Ma 6 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 5.

- **BsmtCond** – ogólny stan piwnicy.
  Ma 6 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 5.

- **BsmtExposure** – ekspozycja piwnicy.
  Ma 5 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 4.

- **BsmtFinType1** – ocena stanu jakości ukończonej części piwnicy.
  Ma 7 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 6.

- **BsmtFinType2** – jest to ocena stanu jakości ukończonej części piwnicy w przypadku gdy dom posiada więcej niż jedną piwnicę.
  Analogicznie jak wyżej, ma 7 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 6.

- **HeatingQC** – jakość i kondycja ogrzewania.
  Ma 5 kategorii porządkowych (od najniższej jakości do najwyższej), którym nadajemy wartości liczbowe od 0 do 4.

- **YearBuilt** – rok budowy.
  Pozostawiliśmy orygianalne wartości.

- **YearRemodAdd** – rok przebudowy, jeśli natomiast jej nie było, to taki sam jak rok budowy.
  Pozostawiliśmy orygianalne wartości.

- **KitchenQual** – jakość kuchni.
  Ma 5 kategorii porządkowych (od najniższej jakości wykonania kuchni do najwyższej), którym nadajemy wartości liczbowe od 0 do 5: Po (Poor), Fa (Fair), TA (Typical/Average), Gd (Good), Ex (Excellent).

- **Functional** – funkcjonalność domu.
  Ma 8 kategorii porządkowych (od najgorszej funkcjonalności do najlepszej), którym nadajemy wartości liczbowe od 0 do 7: Sal, Sev, Maj2, Maj1, Mod, Min2, Min1, Typ.

- **FireplaceQu** – jakość kominka.
  Ma braki danych, które z opisu danych oznaczają brak kominka – nadajemy brakom kategorię 0, pozostałe są uporządkowane od najniższej jakości do najwyższej (od 1 do 5): Po (Poor), Fa (Fair), TA (Typical/Average), Gd (Good), Ex (Excellent).

- **GarageFinish** – stopień wykończenia garażu.
  Ma braki danych, które z opisu danych oznaczają brak garażu – nadajemy brakom kategorię 0, pozostałe są uporządkowane od najmniej wykończonego do najbardziej (kategorie 1-3): Unf (niewykończony), RFn (lekko wykończony), Fin (wykończony).

- **TotRmsAbvGrd** – liczba pokoi jakości dobrej (powyżej normy).
  Od 2 do 14, im więcej tym lepiej.

- **HalfBath** – liczba niepełnych łazienek wykończonych w dobrej jakości.
  Ma wartości liczbowe 0,1,2 i im więcej, tym lepiej.

- **FullBath** – liczba pełnych łazienek wykończonych w dobrej jakości.
  Ma wartości liczbowe 0,1,2,3 i im więcej, tym lepiej.
- **BsmtHalfBath** – liczba niepełnych łazienek w piwnicy wykończonych w dobrej jakości.
  Ma wartości liczbowe 0,1,2 i im więcej, tym lepiej.
- **BsmtHalfBath** – liczba niepełnych łazienek w piwnicy wykończonych w dobrej jakości.
  Ma wartości liczbowe 0,1,2,3 i im więcej, tym lepiej.
- **GarageCars** – pojemność garaża w liczbie aut.
  Ma wartości liczbowe 0,1,2,3 i im więcej, tym lepiej.
- **GarageQual** – jakość garażu.
  Ma braki danych, które z opisu danych oznaczają brak garażu – nadajemy brakom kategorię 0, pozostałe są uporządkowane od najniższej jakości do najwyższej (od 1 do 5): Po (Poor), Fa (Fair), TA (Typical/Average), Gd (Good), Ex (Excellent).
- **GarageCond** – stan garażu.
  Ma braki danych, które z opisu danych oznaczają brak garażu – nadajemy brakom kategorię 0, pozostałe są uporządkowane od najniższej jakości do najwyższej (od 1 do 5): Po (Poor), Fa (Fair), TA (Typical/Average), Gd (Good), Ex (Excellent).
- **PoolQC** – jakość basenu.
  Ma braki danych, które z opisu danych oznaczają brak basenu – nadajemy brakom kategorię 0, pozostałe są uporządkowane od najniższej jakości do najwyższej (od 1 do 5): Po (Poor), Fa (Fair), TA (Typical/Average), Gd (Good), Ex (Excellent).
- **Fence** – jakość ogrodzenia.
  Ma braki danych, które z opisu danych oznaczają brak ogrodzenia – nadajemy brakom kategorię 0, pozostałe są uporządkowane od najniższej jakości do najwyższej (od 1 do 4): MnWw (Minimum Wood/Wire), GdWo (Good Wood), MnPrv (Minimum Privacy), GdPrv (Good Privacy).

2.3 Zmienne ciągłe

- **LotFrontage** – długość ulicy połączonej z posesją. Zastosowano standaryzację.
- **LotArea** – powierzchnia posesji w stopach kwadratowych. Zastosowano standaryzację.
- **MasVnrArea** – powierzchnia obszaru z fornirem murowanym. Zastosowano standaryzację.
- **BsmtFinSF1** – powierzchnia ukończonej piwnicy typu 1. Zastosowano standaryzację.
- **BsmtFinSF2** – powierzchnia ukończonej piwnicy typu 2. Zastosowano standaryzację.
- **BsmtUnfSF** – powierzchnia nieukończonej części piwnicy. Zastosowano standaryzację.
- **TotalBsmtSF** – całkowita powierzchnia piwnicy. Zastosowano standaryzację.
- **1stFlrSF** – powierzchnia pierwszego piętra. Zastosowano standaryzację.
- **2ndflrSF** – powierzchnia drugiego piętra. Zastosowano standaryzację.

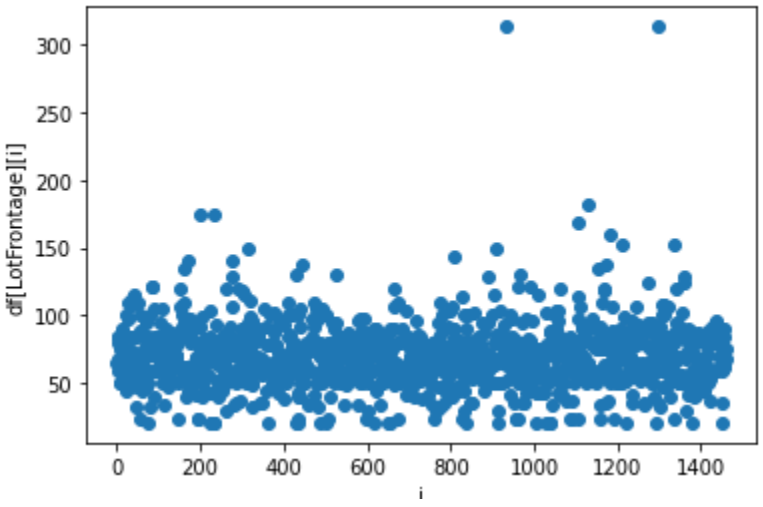
- **LowQualFinSF** – powierzchnia złej jakości. Zastosowano standaryzację.
- **GrLivAre** – powierzchnia dobrej jakości (jakości ponad normę). Zastosowano standaryzację.
- **WoodDeckSF** – drewniana powierzchnia. Zastosowano standaryzację.
- **OpenPorchSF** – powierzchnia werandy na otwartej przestrzeni. Zastosowano standaryzację.
- **EnclosedPorch** – powierzchnia werandy na zamkniętej przestrzeni. Zastosowano standaryzację.
- **MoSold** – miesiąc sprzedaży domu.
- **GarageYrBlt** – rok budowy garażu. Zastosowano standaryzację.
- **3SsnPorch** – powierzchnia werandy przystosowanej do użytku przez 3 pory roku. Zastosowano standaryzację.
- **ScreenPorch** – powierzchnia werandy odgrodzonej szybą. Zastosowano standaryzację.
- **PoolArea** – Powierzchnia basenu. Zastosowano standaryzację.

2.4 Obserwacje odstające

W tej sekcji wypisujemy zmienne podejrzane o posiadanie outlierów, przedstawiamy ich wykresy i wizualnie oceniamy obecność obserwacji odstających. Na podstawie tych wykresów decydujemy w dalszej części projektu o tym, jak traktować wymienione zmienne i ich outliery.

- **LotFrontage**

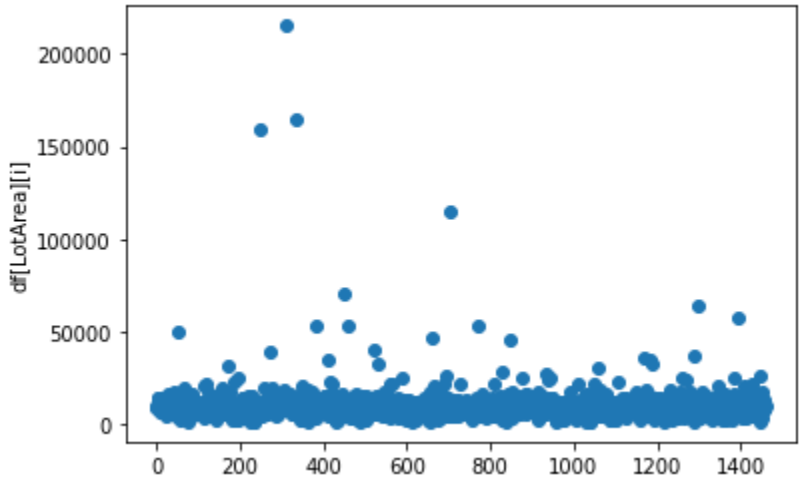
Rysunek 1: Wykres rozproszenia zmiennej LotFrontage



Widzimy tu dwie obserwacje znacząco wystających ponad chmurę punktów.

- **LotArea**

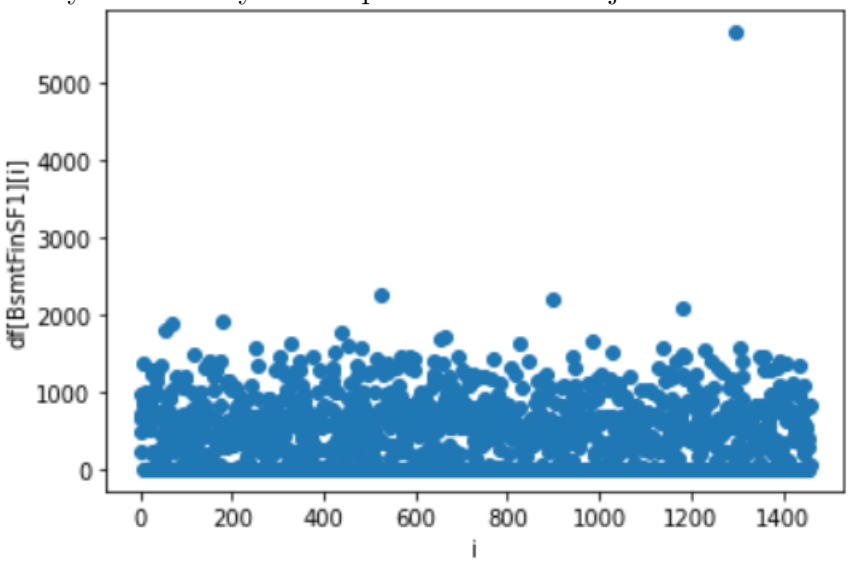
Rysunek 2: Wykres rozproszenia zmiennej LotArea



Widzimy tu kilka obserwacji wystających ponad chmurę punktów.

- **BsmtFinSF1**

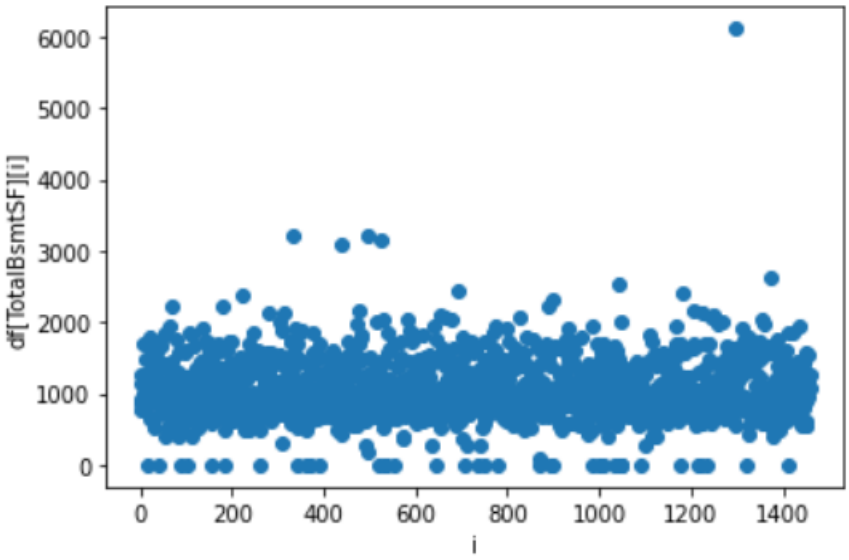
Rysunek 3: Wykres rozproszenia zmiennej BsmtFinSF1



Widzimy tu jedną mocno wyróżniającą się obserwację i trzy delikatnie wystające ponad chmurę punktów.

- **TotalBsmtSF**

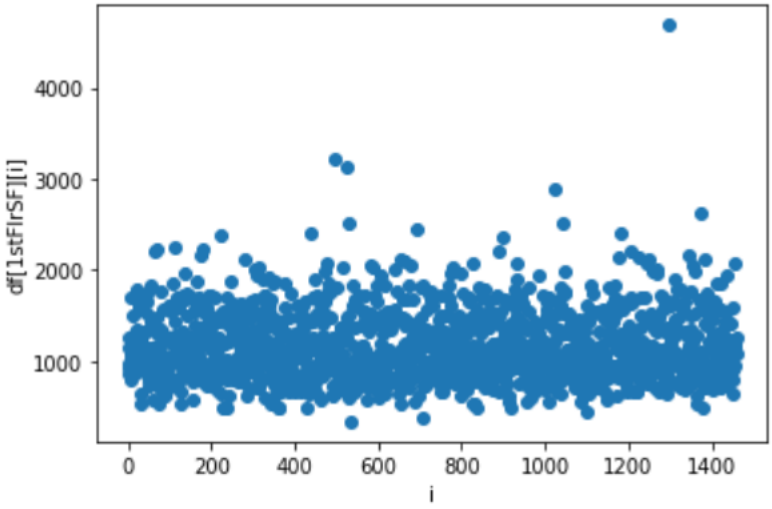
Rysunek 4: Wykres rozproszenia zmiennej TotalBsmtSF



Widzimy tu jedną mocno wyróżniającą się obserwację i cztery delikatnie wystające ponad chmurę punktów.

- **1stFlrSF**

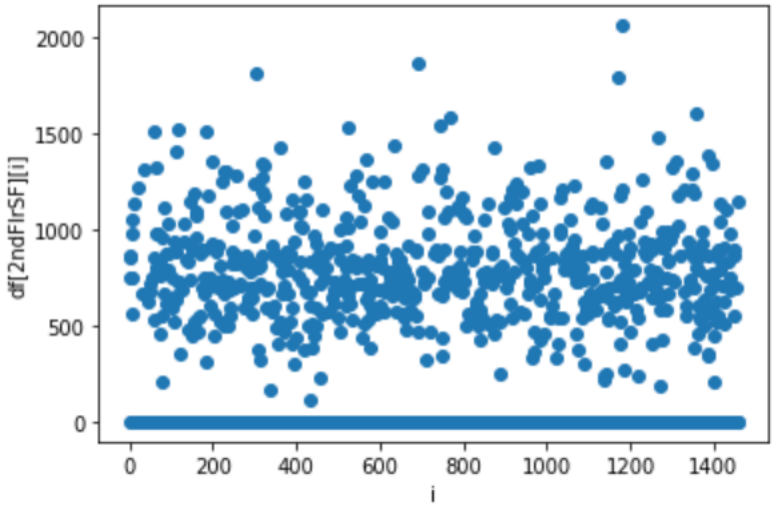
Rysunek 5: Wykres rozproszenia zmiennej 1stFlrSF



Widzimy tu kilka obserwacji wystających ponad chmurę punktów, w tym jedną mocno wyróżniającą się.

- **2ndFlrSF**

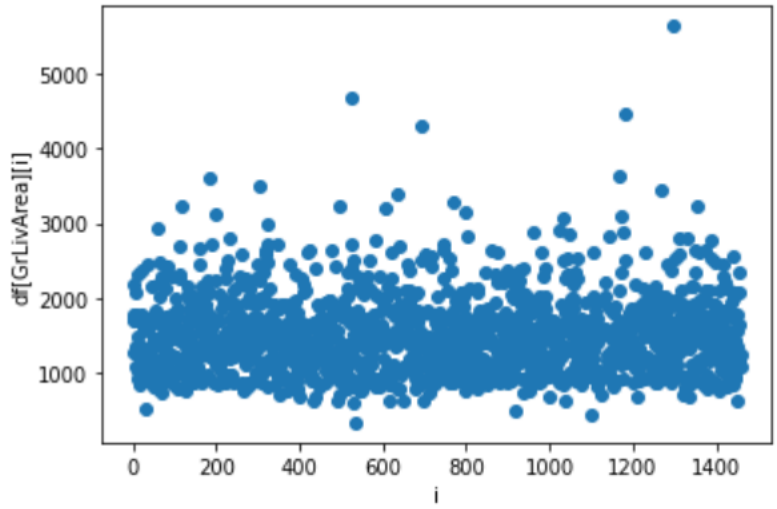
Rysunek 6: Wykres rozproszenia zmiennej 2ndFlrSF



Zakładamy, że zmienna 2ndFlrSF nie ma obserwacji odstających.

- **GrLivAre**

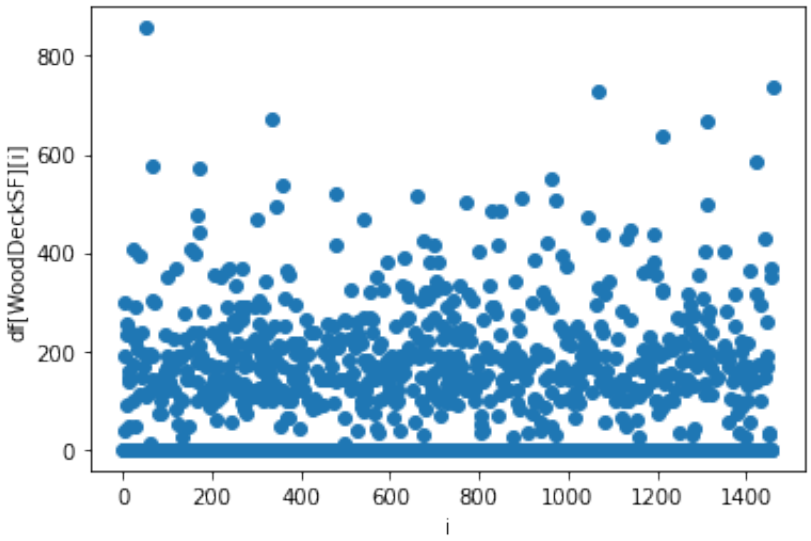
Rysunek 7: Wykres rozproszenia zmiennej GrLivAre



Widzimy kilka wyróżniających się obserwacji.

- **WoodDeckSF**

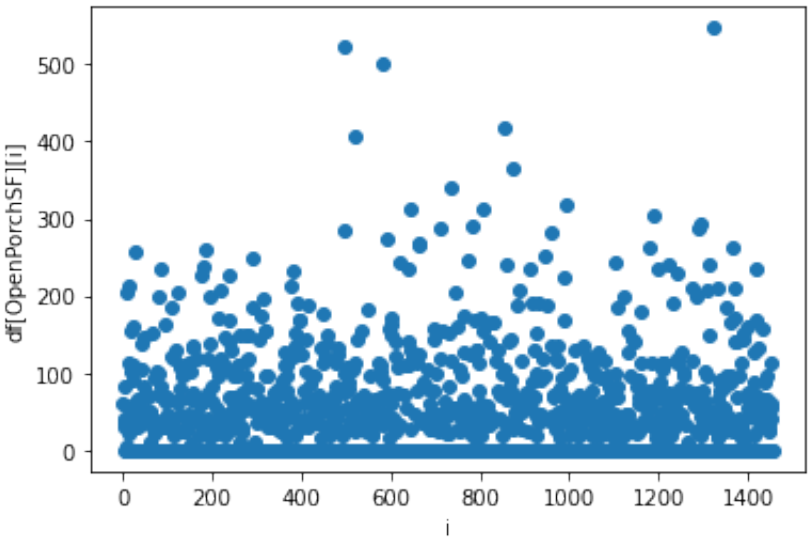
Rysunek 8: Wykres rozproszenia zmiennej WoodDeckSF



Widzimy tu kilka obserwacji wystających ponad chmurę punktów.

- **OpenPorchSF**

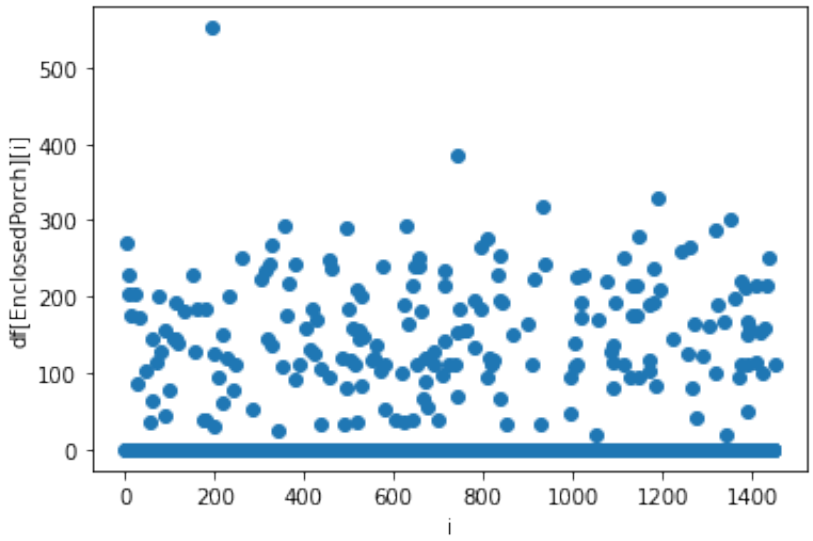
Rysunek 9: Wykres rozproszenia zmiennej OpenPorchSF



Widzimy tu kilka obserwacji wystających ponad chmurę punktów.

- **EnclosedPorch**

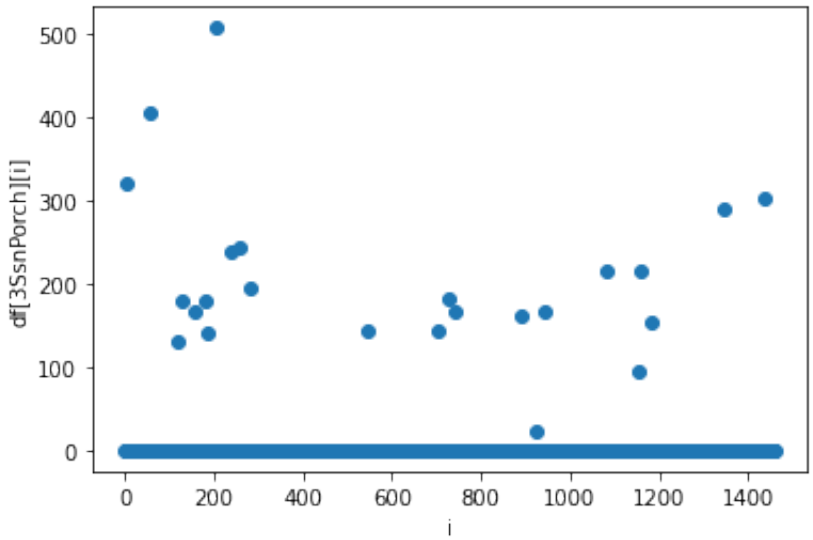
Rysunek 10: Wykres rozproszenia zmiennej EnclosedPorch



Widzimy tu kilka obserwacji wystających ponad chmurę punktów.

- **3SsnPorch**

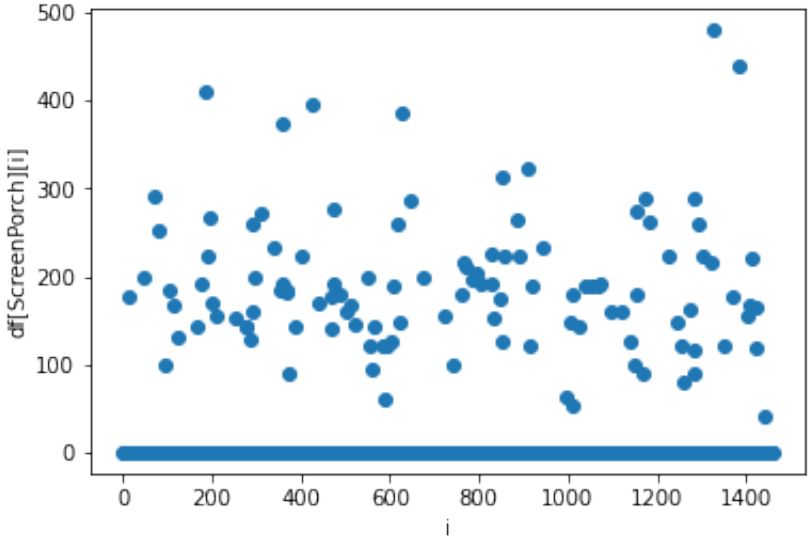
Rysunek 11: Wykres rozproszenia zmiennej 3SsnPorch



Dla tej zmiennej nie ma zbyt wiele danych stąd usuwanie obserwacji odstających nie ma zbyt dużych podstaw.

- **ScreenPorch**

Rysunek 12: Wykres rozproszenia zmiennej ScreenPorch



Zmienna nie posiada wyraźnych obserwacji odstających.

2.5 Selekcja zmiennych

Selekcja zmiennych (ang. feature selection) to proces wyboru pewnego podzbioru ze zbioru wszystkich dostępnych predyktorów (zmiennych objaśniających). Używa się jej np. w celu zmniejszenia czasu obliczeń, uproszczenia struktury modelu czy pozbycia się nieistotnych zmiennych. My postanowiliśmy użyć selekcji zmiennych jeśli poprawiłaby ona (ewentualnie pozostawiła na tym samym poziomie) dokładność, czyli współczynnik R^2, modelu. W tym celu użyliśmy trzech wbudowanych funkcji dostępnych w pakiecie sklearn: PCA, SelectFromModel, SequentialFeatureSelector. Liczba wybranych kolumn dla każdej z funkcji może ulec zmianie po ponownym podziale zbioru na część testową i treningową.

1. **PCA** – Analiza głównych składowych (ang. Principal component analysis) – metoda porządkowania zmiennych w takiej kolejności, aby pierwsza z nich wyjaśniała jak największą zmienność modelu (wariancję) i tak dalej aż do ostatniej, która wyjaśnia najmniej. Załóżmy, że nasz zbiór zmiennych objaśniających ma n kolumn, czyli $X = (X_!, ..., X_n)$. PCA w pierwszym kroku szuka takiego wektora jednostkowego a_1, dla którego projekcja (zmienna losowa) $a_1^T X$ ma największą wariancję, tzn. wychwytuje największą zmienność X. Wtedy $a_1^T X$ to *pierwszy komponent główny*, a znajdowanie go polega na maksymalizowaniu wyrażenia $a^T \mathrm{Cov}(X) a$ przy ograniczeniu $\|a\| = 1$. Każdy kolejny komponent główny jest znajdowany analogicznie i są one w kolejności od tego, który wyjaśnia najwięcej zmienności do tego, który wyjaśnia najmniej. PCA działa najlepiej, gdy zmienne w danych są mocno skorelowane, gdyż komponenty główne są ze sobą nieskorelowane.
   W naszym projekcie wykorzystaliśmy pętle, która buduje model (regresji bayesowskiej) na podstawie wybranych zmiennych odpowiadającym pierwszym i komponentom głównym dla $i = 1, ..., 79$ i dla każdego z nich zapisuje otrzymany współczynnik R^2. Model ostatecznie wybrany przez PCA to model, dla którego otrzymane R^2 było największe. Wybrano 72 kolumny.

2. **SelectFromModel** – jest to funkcja wbudowana w pakiecie sklearn, która wybiera zmienne na podstawie ich wag istotności. Jako argumenty przyjmuje m.in. estymator, czyli model (u nas regresja bayesowska); maksymalną liczbę zmiennych oraz próg, na podstawie którego zmienne są wybierane/odrzucane. Ponieważ my chcieliśmy sprawdzać wyniki dla każdej możliwej liczby wybranych zmiennych, czyli bazować tylko na argumencie max_features, to wartość progu (threshold) należało przyjąć $-\infty$.
   Ponownie sprawdzaliśmy wyniki R^2 za pomocą pętli, która w każdej iteracji zmieniała liczbę zmiennych ($i = 1, ..., 79$) i zapisywała R^2 w liście, z której na końcu wybrano jej maksymalną wartość i odpowiadającą jej liczbę zmiennych. Wybrano 36 kolumn.

3. **SequentialFeatureSelector** – metoda zachłanna (ang. greedy method) wyboru zmiennych do modelu. Ma dwa możliwe kierunki wyboru zmiennych: wprzód, czyli dodawanie zmiennej (przy starcie z najmniejszego modelu) lub wstecz, czyli usuwanie zmiennej (przy starcie z pełnego/największego modelu). Opiera się na optymalizacji lokalnej, czyli wybieraniu zmiennej, która najlepiej poprawia jakość modelu (na podstawie przyjętej odpowiedniej funkcji kryterialnej) w danym momencie, tzn. nie „patrzy w przyszłość" procesu doboru zmiennych.
   Używaliśmy domyślnie ustawionego w funkcji kierunku, czyli forward, a zmienialiśmy ponownie tylko argument odpowiadający za liczbę wybieranych zmiennych do modelu i wybraliśmy te zmienne dające ponownie największy współczynnik R^2. Wybrano 67 kolumn.

Przy PCA napotkaliśmy pewien problem, ponieważ ostatecznie nie udało nam się odzyskać nazw kolumn, które są wykorzystywane przez PCA do tworzenia nowych zmiennych. Według dokumentacji pakietu sklearn taka opcja powinna być dostępna po wywołaniu funkcji *get_feature_names_out()*, jednak w Google Colab wywołanie jej zwracało błąd mówiący o tym, że taka funkcja nie istnieje. Podejrzewamy, że może być to np. problem z wersją Pythona używaną przez Google Colab. Jako że PCA i tak nie było metodą selekcji zmiennych, która dawała najwyższy wynik, to postanowiliśmy z niej zrezygnować i ostatecznie wybierać zmienne za pomocą dwóch pozostałych funkcji. Na ich podstawie stworzyliśmy oddzielne zbiory danych zawierające odpowiednio wybrane kolumny, które w dalszej części projektu posłużą do porównywania działania różnych budowanych przez nas modeli. Problem z działaniem PCA zauważyliśmy dopiero po przetestowaniu modeli na stworzonych zbiorach, kiedy okazało się, że wygenerowany nowy zbiór danych to tak naprawdę pełny model, czyli ze wszystkimi kolumnami. Postanowiliśmy więc go także wykorzystać do porównywania wyników z pozostałymi zbiorami stworzonymi na podstawie dwóch pozostałych funkcji.

3 Budowa modeli

Mieliśmy dostępne na stronie dwa zbiory danych: train.csv oraz test.csv, przy czym tylko pierwszy z nich zawierał także zmienną objaśnianą – cenę domów, wobec tego na nim budowaliśmy i testowaliśmy modele. W tym celu podzieliśmy go na część treningową – 70% obserwacji i testową – 30% obserwacji.

Do wyboru najlepszych parametrów dla każdego modelu używaliśmy funkcji *GridSearchCV*, która przyjmuje parametr *param_grid* – słownik parametrów, z których chcemy wybrać najlepszą kombinację parametrów dla danego modelu i zwraca ten model z dobranymi parametrami.

Aby porównywać otrzymywane rezultaty korzystaliśmy z metody .score dostępnej dla każdego z rozpatrywanych modeli. Metoda ta zwraca współczynnik determinacji R^2 zdefiniowany w następujący sposób:

$$R^2 = \frac{\text{SSR}}{\text{SST}},$$

gdzie
$\text{SSR} = \sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2, \quad \text{SST} = \sum_{i=1}^{n}(Y_i - \bar{Y})^2,$
gdzie Y_i – i-ta obserwacja zmiennej Y, $\hat{Y}_i$ – predykcja i-tej obserwacji na podstawie dopasowanego modelu, $\bar{Y}$ – średnia obserwacji $Y_1, ..., Y_n$.

3.1 Las losowy

3.1.1 Opis modelu

Las losowy (ang. Random Forest) to metoda komitetów/zespołów w uczeniu maszynowym, która polega na konstruowaniu wielu drzew decyzyjnych regresyjnych, a następnie uśrednianiu ich wyniku. Drzewa decyzyjne są najczęściej wybieraną metodą bootstrapu, tzn. ustalamy liczbę B próbek bootsrapowych zbioru treningowego (losowanie ze zwracaniem), dla każdej próbki zbioru, nazwijmy ją D_i budujemy drzewo decyzyjne, które daje wynik M_i. Wtedy ostateczną predykcją będzie

$$\hat{Y} = \frac{1}{B}\sum_{i=1}^{B} M_i.$$

3.1.2 Wybór najlepszych parametrów

Dla funkcji *RandomForestRegressor* wybieraliśmy najlepsze z następujących parametrów:

- n_estimators – liczba drzew w lesie losowym, z przedziału [10,1010] co 50,
- min_samples_split – minimalna liczba obserwacji potrzebna, aby rozdzielić liść drzewa na grupy, z przedziału [2,20] co 1.

Najlepsze parametry: n_estimators = 110, min_samples_split = 7.

3.1.3 Wyniki

$R^2 \approx 0.876$

3.2 Bagging

3.2.1 Opis modelu

Bagging to metoda komitetów/zespołów w uczeniu maszynowym, która polega na dopasowywaniu składowych modeli na losowych podzbiorach oryginalnego zbioru danych, a następnie uśrednianiu ich wyników w celu osiągnięcia końcowego rezultatu. Domyślnie, jako składowe modele używane są drzewa decyzyjne.

3.2.2 Wybór najlepszych parametrów

Dla funkcji *BaggingRegressor* wybieraliśmy najlepsze z następujących parametrów:

- n_estimators – liczba drzew decyzyjnych, z przedziału [10,150] co 10,
- max_samples – ilość próbek ze zbioru do trenowania pojedynczego modelu, z przedziału [30,90] co 3.

Najlepsze parametry: n_estimators = 70, max_samples = 87.

3.2.3 Wyniki

$R^2 \approx 0.847$

3.3 Regresja liniowa

3.3.1 Opis modelu

Regresja liniowa jest jedną z najbardziej standardowych metod używanych w teorii uczenia maszynowego. Zakłada ona, że pomiędzy zmienną objaśnianą (odpowiedzią) $\mathbf{Y}$ a zmiennymi objaśniającymi (predyktorami) $\mathbf{X}_i$. Zależność ta jest modelowana z uwzględnieniem błędu losowego ε. Zatem model liniowy ma postać:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

gdzie $\mathbf{Y} \in \mathbb{R}^n$, $\mathbf{X} \in \mathbb{R}^{n \times p}$ – macierz, której kolumny są predyktorami, $\boldsymbol{\beta} \in \mathbb{R}^p$ - wektor współczynników kombinacji liniowej, a $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ – wektor niezależnych zmiennych losowych. Zadaniem, które rozwiązujemy jest przewidywanie zmiennej $\mathbf{Y}$ na podstawie macierzy $\mathbf{X}$ poprzez znajdowanie współczynników $\boldsymbol{\beta}$, które zapewniają jak najlepsze dopasowanie modelu. Najbardziej standardowym podejściem (używanym także w pakiecie sklearn, z którego korzystamy) jest metoda najmniejszych kwadratów, która minimalizuje sumę kwadratów błędów popełnianych przy predykcji każdego Y_i dla $i = 1, ..., n$, tzn.

$$\boldsymbol{\beta}_{MNK} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\arg\min} \, \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 = \sum_{i=1}^{n} \left(Y_i - \sum_{j=1}^{p} X_{ij}\beta_j \right)^2 .$$

3.3.2 Wybór najlepszych parametrów

Funkcja *LinearRegression* nie posiada parametrów, z których można byłoby wybierać najlepsze. Wobec tego zostawiamy ją z domyślnymi parametrami.

3.3.3 Wyniki

$R^2 \approx 0.8868$

3.4 Regresja bayesowska

3.4.1 Opis modelu

Bayesowka regresja liniowa jest statystycznym podejściem do regresji liniowej w którym wykorzystywane jest wnioskowanie Bayesowskie. Zakładamy, że błędy są od siebie niezależne i mają rozkład normalny. Dodatkowo jako rozkład apriori dla parametrów alpha i lambda bierzemy rozkład Gamma. Razem z liczbą iteracji poddajemy je testom, dzięki którym możemy wybrać ich najlepsze wartości.

3.4.2 Wybór najlepszych parametrów

- n_iter – maksymalna liczba iteracji, ze zbioru [1, 5, 10, 20, 30, 50, 100, 300],
- alpha_1 – parametr kształtu dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-8, 1e-6, 1e-4].
- alpha_2 – odwrócony parametr skali dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-8, 1e-6, 1e-4, 1e-2, 1].

- lambda_1 – parametr kształtu dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-8, 1e-6, 1e-4, 1e-2, 1].
- lambda_2 –odwrócony parametr skali dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-8, 1e-6, 1e-4].

Najlepsze parametry: n_iter = 10, alpha_1 = 1e-08, alpha_2 = 1, lambda_1 = 1, lambda_2 = 1e-08

3.4.3 Wyniki

$R^2 \approx 0.8874$

3.5 Regresja ARD

3.5.1 Opis modelu

Regresja ARD (Automatic Relevance Determination) działa podobnie do regresji Bayesowskiejm jednak w przeciwieństwie do niej zamiast używać zwykłej metody najmniejszych kwadratów skaluje otrzymane współczynniki w stronę zera co zapewnia ich większą stabilność. Podobnie jak w przypadku regresji Bayesowkiej zajmujemy się parametrami rozkładu apriori Gamma oraz liczbą iteracji.

3.5.2 Wybór najlepszych parametrów

- n_iter – maksymalna liczba iteracji, ze zbioru [1, 5, 10, 30, 50, 100],
- alpha_1 – parametr kształtu dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-8, 1e-6, 1e-4, 1e-2, 1].
- alpha_2 – odwrócony parametr skali dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-6, 1e-4, 1e-2, 1].
- lambda_1 – parametr kształtu dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-6, 1e-4, 1e-2, 1].
- lambda_2 –odwrócony parametr skali dla rozkładu Gamma apriori dla parametrem alfa, ze zbioru [1e-6, 1e-4, 1e-2, 1].

Najlepsze parametry: n_iter = 10, alpha_1 = 1e-08, alpha_2 = 1, lambda_1 = 1e-2, lambda_2 = 1e-04

3.5.3 Wyniki

$R^2 \approx 0.8868$

3.6 Boosting

3.6.1 Opis modelu

Podstawowym algorytmem do implementacji boostingu jest AdaBoost (ang. Adaptive Boosting). Podobnie jak las losowy jest to metoda komitetów, czyli polega na iteracyjnym budowaniu wielu modeli. Za każdym razem gdy dopasowywany jest model, algorytm przypisuje większe wagi źle sklasyfikowanym obserwacjom, dzięki czemu kolejne modele "zwracają większą uwagę" na te obserwacje.

Wagi rekordów są wybierane w taki sposób, aby błąd na zbiorze treningowym malał z szybkością wykładniczą. Istotną cechą boostingu jest to, że przy przypisywaniu wag zapominamy o poprzednim modelu, więc kolejne modele są od siebie niezależne, a to skutkuje większą zmienniczością modelu.

3.6.2 Wybór najlepszych parametrów

Najczęściej wykorzystywanym modelem w algorytmie AdaBoost jest drzewko decyzyjne i dlatego też funkcja *AdaBoostRegressor* ma ustawiony model *DecisionTreeRegressor* jako domyślny. Oprócz drzewek spróbowaliśmy do niego dopasować także kilka innych modeli. Dla każdego przypadku dobieraliśmy najlepszą liczbę modeli z przedziału [10, 1010] co 10.

- Drzewka decyzyjne: n_estimators = 800,
- Regresja liniowa: n_estimators = 10,
- Regresja bayesowska: n_estimators = 460,
- Regresja grzbietowa: n_estimators = 10,
- Regresja ARD: n_estimators = 10.

3.6.3 Wyniki

Selekcja zmiennych	Outliering	Regresja liniowa	Las losowy	Bagging	Regresja bayesowska	Regresja ARD	AdaBoost*
brak	brak	0.829	0.876	0.847	0.878	0.874	0.749
brak	usunięcie	0.848	0.854	0.815	0.879	0.872	0.842
brak	uśrednienie	0.702	0.806	0.783	0.709	0.707	0.770
SFM	usunięcie	0.859	0.843	0.819	0.859	0.859	0.835
SFM	uśrednienie	0.887	0.865	0.836	0.887	0.887	0.829
SFS	usunięcie	0.865	0.862	0.829	0.872	0.865	0.836
SFS	uśrednienie	0.806	0.862	0.828	0.810	0.798	0.865

* – model AdaBoost z najlepszym prostym modelem regresji dla danego zbioru danych.
Poniżej przedstawione zostały najlepsze wyniki dla poszczególnych klasyfikatorów:

- Drzewka decyzyjne: $R^2 \approx 0.8760$,
- Regresja liniowa: $R^2 \approx 0.8868$,
- Regresja bayesowska: $R^2 \approx 0.8874$,
- Regresja grzbietowa: $R^2 \approx 0.8470$,
- Regresja ARD: $R^2 \approx 0.8868$.

Poniżej przedstawione zostały najlepsze wyniki dla poszczególnych modyfikacji zbioru danych:

- brak selekcji, brak outlieringu: $R^2 \approx 0.878$,
- brak selekcji, usunięcie outlierów: $R^2 \approx 0.879$,
- brak selekcji, uśrednienie outlierów: $R^2 \approx 0.806$,

- SFM, usunięcie outlierów: $R^2 \approx 0.859$,
- SFM, uśrednienie outlierów: $R^2 \approx 0.887$,
- SFS, usunięcie outlierów: $R^2 \approx 0.872$,
- SFS, uśrednienie outlierów: $R^2 \approx 0.865$.

4 Podsumowanie

Na sam początek podsumowania zauważmy, że wszystkie rozważane modele po dobraniu optymalnych parametrów uzyskały bardzo dobre wyniki, z pojedynczymi wyjątkami z wynikiem $R^2 \approx 0.7$. Zacznijmy od selekcji zmiennych i sposobu radzenia sobie z outlierami. Ewidentnie najgorsza możliwa kombinacja to brak selekcji zmiennych połączony z wstawieniem średniej w miejsce wartości odstających, co nie do końca jest intuicyjne. Można by pomyśleć, że gdy zostawimy outliery to wynik będzie jeszcze gorszy, ale nasze testy pokazały, że wstawienie w tej konkretnej sytuacji średniej pogarsza ogólny wynik każdego z estymatorów - każdy estymator z wyłączeniem AdaBoost'a miał najgorszy wynik właśnie na tym zbiorze.

Powiedzieliśmy już o najgorszych zbiorach, więc teraz czas na najlepsze. Kombinacja, która dawała najlepsze wyniki, i tutaj znów musimy zaznaczyć, że niezależnie od doboru estymatora, to selekcja zmiennych metodą SFM i uśrednienie wartości odstających. Mówiąc już konkretnie, najlepszy wynik dała regresja bayesowska: $R^2 \approx 0.8874$.

Wnioski:

- Najlepiej radziły sobie estymatory regresji liniowej, ze szczegolnym naciskiem na regresję bayesowską i ARD. Nie powinno nas to dziwić, bo w zadaniu estymacji cen domu na podstawie jego cech regresja sprawdza się bardzo dobrze.
- Las losowy zwracał zaskakująco dobre wyniki. Myśleliśmy, że las losowy będzie nieco odstawał, ale różnił się nieznacznie od najlepszych estymatorów.
- AdaBoost wypadł znacznie gorzej niż zakładaliśmy.
- Przy selekcji zmiennych SFS, która ucieła znacznie mniej kolumn niż SFM, estymatory las losowy i AdaBoost radziły sobie wyraźnie lepiej niż reszta.
- Wyniki na zbiorze bez selekcji zmiennych i bez usunięcia outlierów były niewiele gorsze niż na najlepszym zbiorze.

5 Opis programu

Cały kod projektu zamieszczony jest w pliku jupyter notebook, który podzielony jest na pięć części:

- Data understanding,
- Data preparation,
- Feature selection,
- Modeling.

Odpowiadają one kolejnym etapom procesu tworzenia modelu uczenia maszynowego. Ta sekcja poświęcona jest opisowi kodu znajdującego się w każdej z tych części.

5.1 Data understanding

Pierwsze linijki kodu odpowiadają za import wszystkich potrzebnych pakietów oraz wgranie, a następnie wyświetlenie danych. W celu głębszej analizy 79 zmiennych objaśniających został stworzony skrypt wypisujący następujące informacje o każdym predyktorze po kolei.

- Informacja o tym czy kolumna posiada wartości NaN.
- Lista wszystkich unikatowych wartości znajdujące się w danej kolumnie wraz z informacją ile ich łącznie jest.
- Wykres rozproszenia pozwalający stwierdzić obecność wartości odstających.

Pod wymienionym wyżej skryptem znajduje się jeszcze jeden, który odpowiada za podsumowanie uzyskanych danych. Za pomocą tych informacji można określić strategię, według której w kolejnej części kodu dane zawierające zmienne objaśniające będą przygotowane pod użycie w modelach.

5.2 Data preparation

W części zajmującej się przygotowaniem danych dla modeli, na podstawie wcześniejszych analiz, dla każdej z kolumn pozbyliśmy się poprzez wypełnienie odpowiednią wartością wierszy zawierających brak danych. Dla zmiennych kategorycznych użyliśmy zamiany kategorii na liczby z użyciem predefiniowanych słowników. Natomiast dla zmiennych jakościowych, które nie były kategoryczne wykorzystaliśmy wbudowaną w pakiet sklearn funkcję LabelEncoder. Z tego samego pakietu funkcję StandardScaler użyliśmy do przeskalowania zmiennych jakościowych. Na końcu jeżeli w danej kolumnie występowały outliery, jako outliery uważaliśmy wartości odbiegające o 4 wartości odchylenia standardowego od średniej kolumny, to postępowaliśmy zgodnie z jedną z 3 metod. Usunięcie wierszy, zastąpienie wartości modą albo pozostawienie outlierów. W ten sposób wygenerowaliśmy różne zbiory danych, na których potem mogliśmy zbudować modele i porównać ich wyniki.

5.3 Feature selection

Do selekcji zmiennych użyliśmy ostatecznie dwóch funkcji opisanych w sekcji 2.5, czyli *SelectFromModel* oraz *SequentialFeatureSelector*. Obie z nich wybrały "swój" najlepszy (tzn. dający najwyższy score) podzbiór zmiennych całego modelu. Wybory obu funkcji zapisaliśmy w oddzielnych listach, na podstawie których stworzyliśmy nowe zbiory danych (pliki .csv) użyte następnie do testowania i porównywania modeli, oprócz nich pozostawiliśmy jako oddzielny zbiór danych ten, który zawierał wszystkie kolumny, aby z nim także porównywać otrzymywane wyniki.

5.4 Modeling

Każdy zbiór danych, wygenerowany w sposób opisany powyżej, został losowo podzielony na zbiór uczący i zbiór testowy w proporcjach 70%/30%. Dla każdego zbioru uczącego uruchomiliśmy dopasowywanie podstawowych klasyfikatorów:

- regresji liniowej,

- lasu losowego,
- baggingu,
- regresji ARD,
- regresji bayesowskiej.

W tym momencie również za pomocą algorytmu *grid search* znaleźliśmy najlepsze hiperparametry tych klasyfikatorów dla konkretnych zbiorów danych. Modele zostały również zserializowane do plików, aby następnie można było z nich skorzystać ponownie w łatwy sposób.

Następnie dla najlepszego modelu prostego dla każdego zbioru danych, uruchomiliśmy model AdaBoost, również dopasowując najlepsze hiperparametry automatycznie z wykorzystaniem *grid searcha*.

Walidację zbudowanych modeli przeprowadziliśmy wykorzystując funkcję score wywołaną na danych testowych, która liczy współczynnik determinacji (R kwadrat).