Raport

Łukasz Fabia

20.05.2024

Spis treści

1 Wstęp

Celem badań jest analiza danych dotyczących ofert pracy w IT. W swojej pracy postaram się odpowiedzieć na pytanie, jakie są najbardziej poszukiwane umiejętności w branży IT oraz ile można zarobić znając dane języki, frameworki czy narzędzia. W tym celu postram się wykorzystać sci-kit learn do stworzenia modelu regresji liniowej, który pozwoli mi przewidzieć zarobki na podstawie umiejętności (technologii).

2 Dane

Dane pozyskam z serwisu justjoin.it, który zbiera oferty pracy z wielu różnych serwisów, zatem ofert pracy będzie całkiem sporo. Na stronie mamy katergorie, które mogą być przydatne do analizy, takie jak: JS, PHP, Ruby, Python, Java, Net, Mobile, C, DevOps, Security, Data, Go, Game, Scala. W mojej analizie skupię się na nich. Dodatkowo analizuję zarobki tylko na b2b oraz na umowie o prace (uop), ponieważ są to najbardziej popularne formy zatrudnienia w IT a inne formy takie jak umowa o zlecenie czy umowa o staż pratykcznie nie występują. Do analizy będę również brał pod uwagę lokalizację.

Technologia - język programowania, framework, narzędzie, które jest wymagane w ofercie pracy.

2.1 Model danej

Dane będą zawierały informacje o ofertach pracy, takie jak:

- tytuł oferty
- widełki dla B2B
- widełki dla UOP
- technologie dotyczące umowy
- lokalizacja
- doświadczenie junior, mid, senior
- typ pracy stacjonarnie, hybrydowo, zdalnie

2.2 Obsługa technologii, lokalizacji

Najpierw zdefiniuje sobie słownik klucz, wartość, gdzie klucz to ustandaryzowana technologia, a wartość do synonimy tej technologii.

np. "JavaScript": ["javascript", "js", "node.js", "nodejs", "express.js", "expressjs",],

Dzięki temu będę mógł przekonwertować technologie z oferty pracy na wektor binarny, gdzie 1 oznacza, że technologia jest wymagana, a 0, że nie jest wymagana. Kolejnym krokiem będzie obsługa lokalizacji. W tym przypadku jeśli oferta dot. kilku miast to znaczy, że pojawi się w zbiorze klika ofert z tymi samymi danymi, ale dla różnych miast.

2.3 Pozykiwanie danych

Dane będą pozyskiwane z ww. serwisu, za pomocą narzędzi do web scrappingu w moim przypadku będzie to `Selenium`, ponieważ strona ma dynamicznie ładowany content.

Kroki:

- napisanie skryptu pobierającego linki do ofert pracy z danej kategorii, ponieważ nie chcemy śmiecowych ofert typu Product manager
- napisanie skryptu przetwarzającego linki do ofert pracy, aby pobrać dane z oferty
- przekierowanie wyniku do pliku json.
- normalizacja oraz oczyszczanie danych, kodowanie technologii, do wektora przy pomocy MultiLabelBinarizer z `sklearn`
- kodowanie duplkacja ofert z różnymi lokalizacjami oraz kodowanie typu pracy i doświadczenia (`label encoding`)
- usunięcie ofert z wynagrodzniem godzinowym bo zależą one od ilości przepracowanych godzin

Ofert ze stawką godzinową było kilka więc nie wypływają one na wyniki.

3 Wygląd do danych

uwaga przykładowe dane nie zawierają wszystkich kolumn bo jest ich za dużo, wszystkie dane można znaleźć w ../data/jobs.csv

Przykładowe dane:

title	min_b2b	max_b2b	min_uop	max_uop
Senior Software Engineer	0.0	0.0	18000.0	28000.0
Senior Backend Node.js Engineer	0.0	0.0	18360.0	25125.0
Senior Fullstack Developer	22680.0	27216.0	16600.0	19920.0

location_code	operating_mode_code	experience_code
38	0	2
17	2	2
51	0	2

AWS	JavaScript	React	Java
1	1	1	0
0	1	1	0
1	1	1	0

4 Rozkłady i statystyki

Aktualnie w zbiorze *jobs.csv* znajduje się **4574** ofert pracy, które będą poddane analizie. Wszystkie dane są znormalizowane i gotowe do analizy. Analizę można zacząć od średniej zarobków dla kontraktu B2B oraz UOP.

Widełki dla Juniora:

PLN	B2B	UOP
średnie widełki	8555.40	13558.71
min widełki	4250.00	6000.00
max widełki	16443.00	28000.00

Tabela 1: Średnie zarobki w PLN dla `juniora` w Polsce

Widełki dla Mida:

PLN	B2B	UOP
średnie widełki	12378.99	18041.77
min widełki	5000.00	7000.00
max widełki	25000.00	30000.00

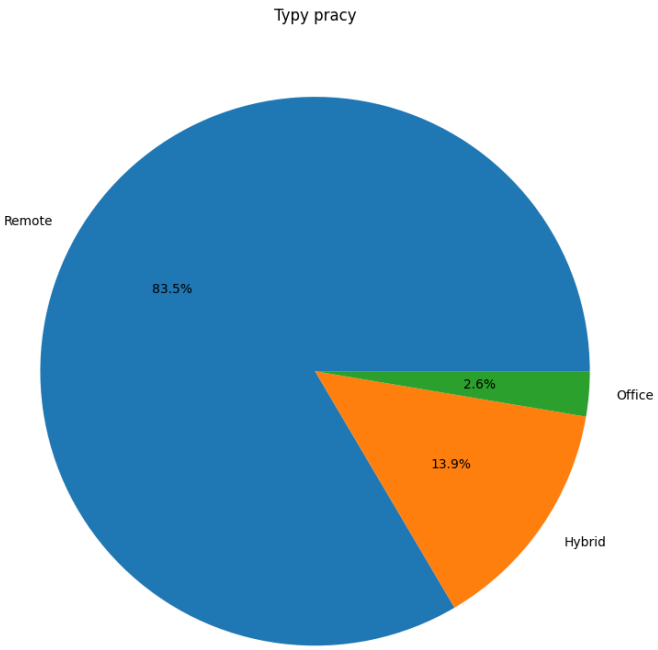
Tabela 2: Średnie zarobki w PLN dla `mida` w Polsce

Widełki dla Seniora:

PLN	B2B	UOP
średnie widełki	18930.61	25848.46
min widełki	8000.00	11000.00
max widełki	40000.00	80000.00

Tabela 3: Średnie zarobki w PLN dla `seniora` w Polsce

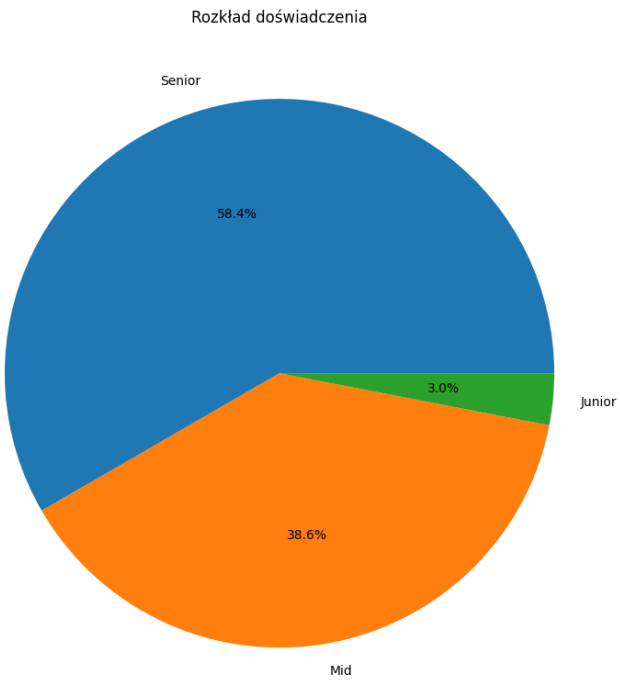
4.1 Jak się pracuje w IT?



Rysunek 1: Rozkład typów pracy

Jak widać najwięcej ofert pracy dotyczy pracy zdalnej.

4.2 Kogo szukają pracodawcy?



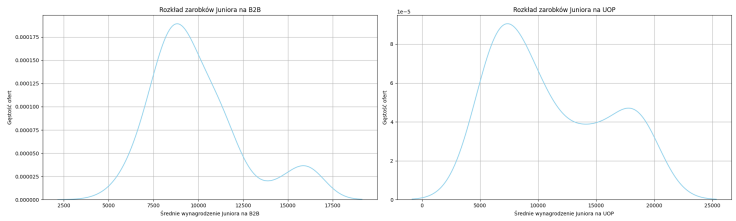
Rysunek 2: Rozkład typów pracy

Tak jak można było się spodziewać - najwięcej ofert pracy jest dla seniorów, stąd też wynika dlaczego tak dużo kontraktów dotyczy pracy zdalnej. Chociaż warto powiedzieć sytuacja midów jest również dobra. Gorzej jest z ofertami dla młodych programistów. Tutaj liczba ofert wyniosła zaledwie 139, co jest bardzo małą liczbą w porównaniu do innych grup.

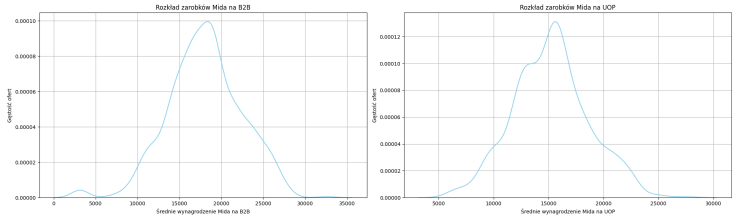
Czy to oznacza, że młodzi programiści mają trudniej, a słynne "eldorado" w IT jest tylko dla doświadczonych programistów?

Tutaj można powiedzieć, że juniorzy mają trudniej **wejść** do branży, ale zarobki po wejściu są naprawdę atrakcyjne, no, ale tutaj problem może być z wejściem.

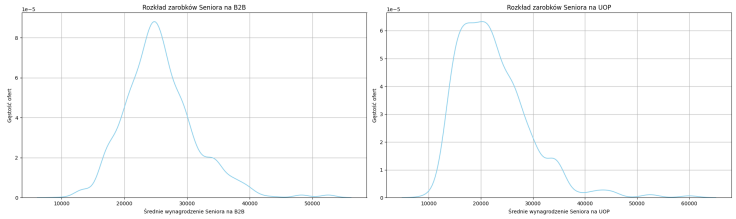
4.3 Jak rozkładają się zarobki?



Rysunek 3: Rozkłady zarobków dla poszczególnych umów dla juniorów

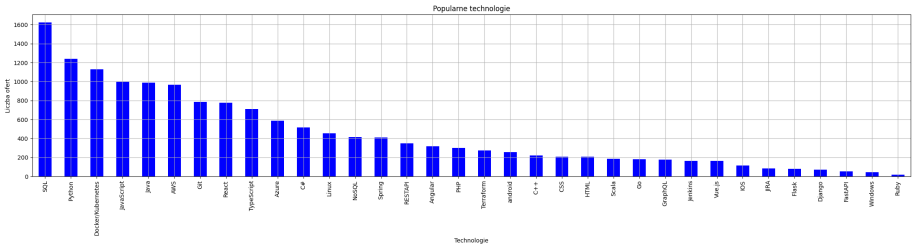


Rysunek 4: Rozkłady zarobków dla poszczególnych umów dla midów



Rysunek 5: Rozkłady zarobków dla poszczególnych umów dla seniorów

4.4 Jakie technologie są najbardziej poszukiwane?

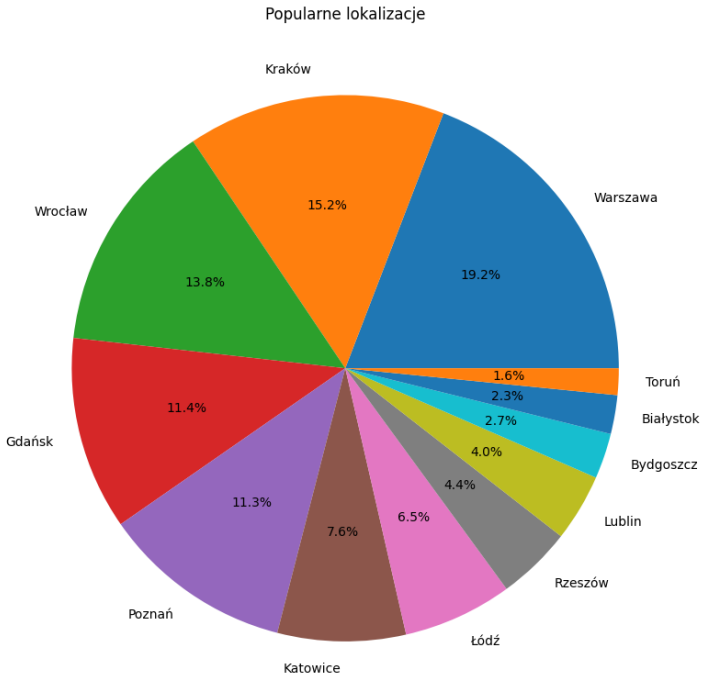


Rysunek 6: Popularne technologie w ofertach pracy w Polsce

Tutaj moim zdaniem troche zaskoczenie ponieważ bez `SQL` ciężko znaleźć prace w IT, czyli bazy danych to jest podstawa przy rekrutowanu się do pracy.

Oczywiście nie mogło zabraknąć `Pythona` oraz `JavaScriptu` jeśli chodzi o języki skryptowe. Co warto zazanczyć narzędzia takie jak `Docker` czy `Kubernetes` również są bardzo popularne i warto je znać. `Java` wygrywa z `C#` a `GNU/Linux` deklasuje `Windowsa`.

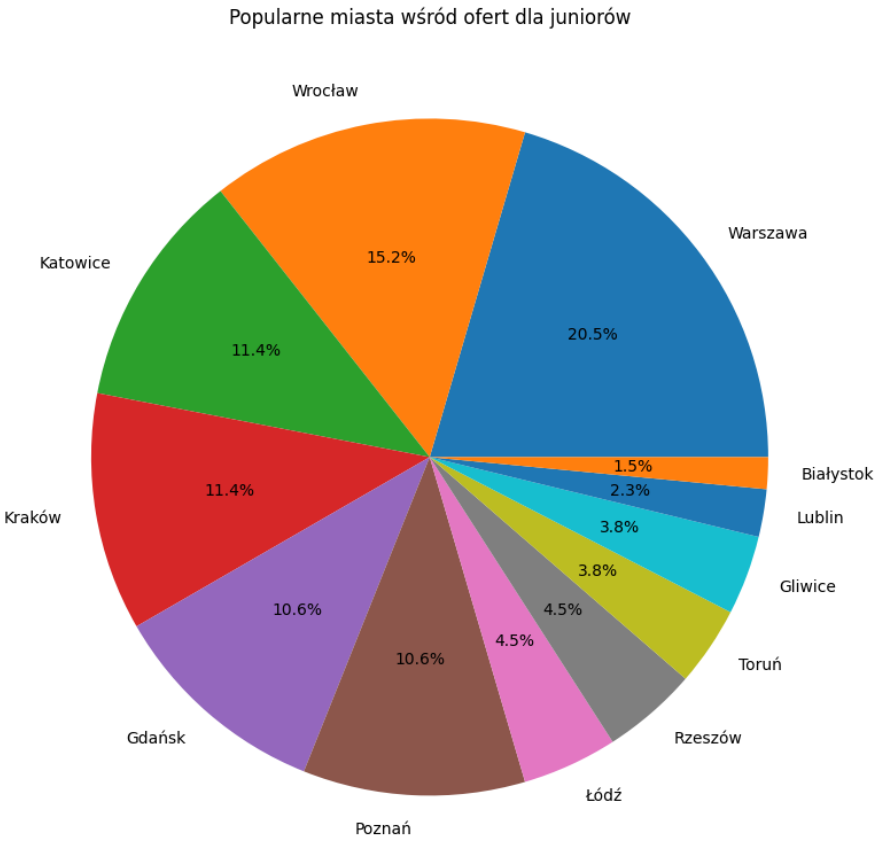
4.5 Gdzie jest największy popyt na programistów?



Rysunek 7: Popularne miasta w ofertach pracy w Polsce

Zestawienie miast jest zgodne z oczekiwaniami, najwięcej ofert pracy jest kolejno w **Warszawie**, **Krakowie** oraz **Wrocławiu**, chociaż **Gdańsk** również pojawiał się w dużej ilości ofert pracy.

4.6 Gdzie poszukiwani są juniorzy?

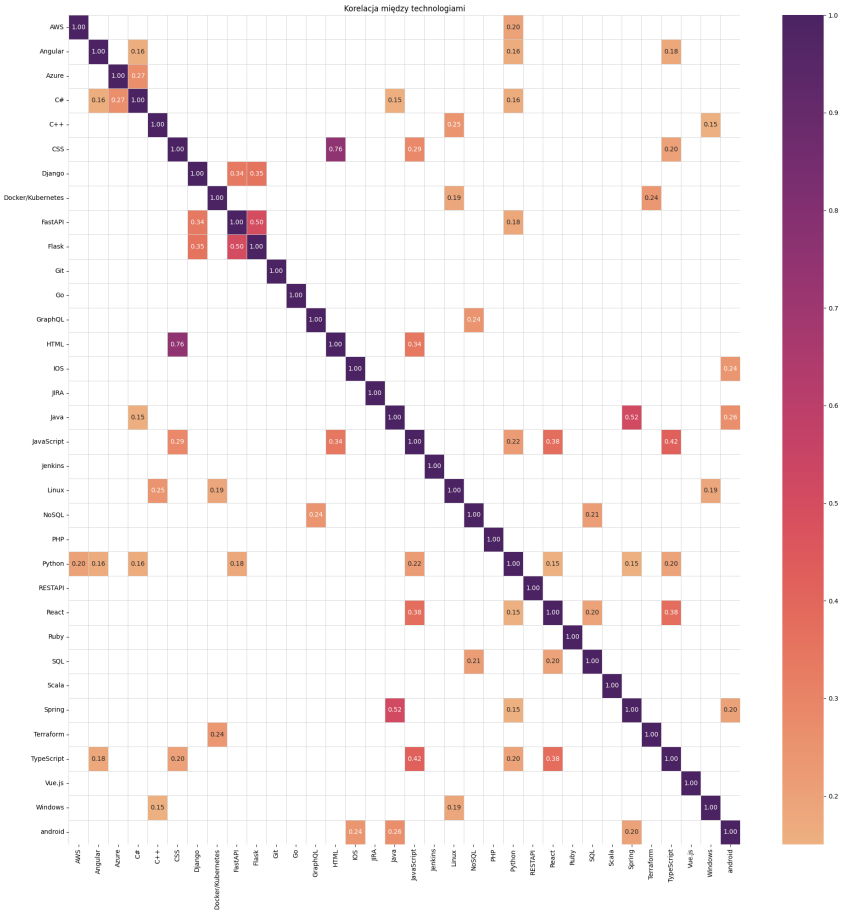


Rysunek 8: Popularne miasta w ofertach dla juniorów

Warszawa jest najbardziej przyjazna dla juniorów, ale warto zauważyć, że wykres nie różni się bardzo od poprzedniego z jednym, *ale* - **Katowice** są na 3 miejscu w zestawieniu dla juniorów, co może być zaskoczeniem.

5 Powiązania między danymi

5.1 Powiązania między technologiami



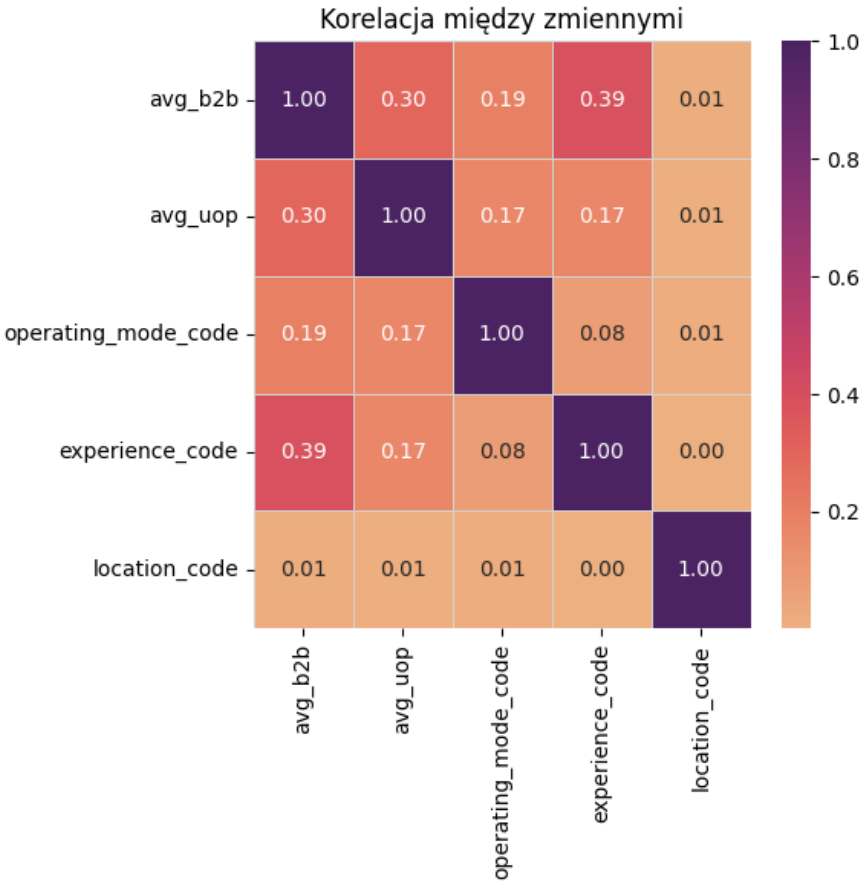
Rysunek 9: Powiązania między technologiami, zawierająca tylko wartości korelacji większe niż 0.14

Co można zauważyć?

1. HTML i CSS idą ze prawie w parze - co jest zrozumiałe, bo to podstawy front-endu
2. Przy Javie warto znać Springa
3. React i JS i TS często pojawiają sie razem w ofertach pracy obok HTML i CSS
4. Jak sie uczy Django to warto znać inne frameworki backendowe takie jak Flask czy FastAPI
5. Jak sie idzie w Embedded to warto znać C/C++ oraz Linux

To tylko kilka przykładów wymienionych wynikający z obrazka powyżej, ale warto zauważyć, że nie ma tutaj dużo powiązań między technologiami, co może wynikać z tego, że technologie są zbyt różne, aby były powiązane.

5.2 Powiązania między innymi zmiennymi

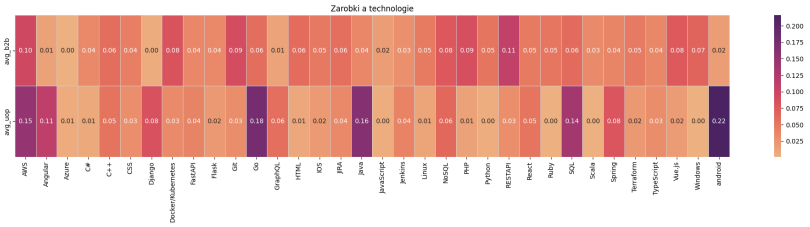


Rysunek 10: Powiązania między innymi zmiennymi

Co można zauważyć?

1. W jakiś spobób powiązane są ze sobą zarobki na B2B i UOP - ma sens
2. Wynagrodzenie na B2B i UOP jest powiązane z doświadczeniem

5.3 Zarobek a technologie



Rysunek 11: Powiązania między zarobkiem a technologiami

Tutaj jest kilka ciekawych powiązań, które warto zauważyć, np. na umowie o prace znaczenie ma znajomość: Go, AWS, Angulara, Java, SQL czy Andorida, chociaż nie są to mocne powiązania. Natomiast na B2B nie ma jakiś znaczących powiązań można wskazać np. Resta, AWS, Docker/Kubernetes czy PHP, ale są to watości rzędu 0.09, co nie jest imponującym wynikiem.

6 Czy da się przewidzieć zarobki, w zależności od mojego tech-stacku?

6.1 Ogólnie o problemie

Oczywiście, że tak, kiedy mamy dane to możemy nauczyć model, który na wejściu dostanie zmienne i przewidzi dla nas zarobki. Dokładniej mówiąc model otrzyma na wejściu dane takie jak:

`Input:`

- Tech-stack

`Output:`

- Zarobki w PLN

6.2 Dobór modeli

Modele które będą wykorzystane w analizie to:

1. Regresja liniowa
    - `LinearRegression`
    - `Ridge`
    - `Lasso`
    - `ElasticNet`
2. `Decision tree`
3. `Random forest`

Wszyskie modele pochodzą z modułu ***`sklearn`*** *dostępniej pod tym linkiem*

6.3 Trochę statystyki - metryki

Do oceny modeli wykorzystam metryki takie jak:

- **Root Mean Squared Error** - pierwiastek z średniego błędu kwadratowego
- **R-squared** - współczynnik determinacji R^2
- **Mean Absolute Error** - średni błąd bezwzględny

6.3.1 Pierwiastek z średniego błędu kwadratowego

Root Mean Squared Error (RMSE) - to pierwiastek z MSE, co daje nam miarę błędu przewidywań w tych samych jednostkach co dane wejściowe. Jest bardziej intuicyjny w interpretacji niż MSE.

6.3.2 Współczynnik determinacji

R-squared (R2) - to miara oceny dopasowania funkcji regresji do danych. Wartość bliska 1 oznacza, że funkcja regresji lepiej dopasowała sie do danych.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y_i})^2}{\sum(y_i - \bar{y})^2}, R^2 \in [0, 1] \tag{1}$$

6.3.3 Średni błąd bezwzględny

Mean Absolute Error (MAE) - to średni bezwzględny błąd między przewidywaniami a rzeczywistymi wartościami. MAE mierzy średnią wielkość błędów w przewidywaniach modelu, nie zwracając uwagi na kierunek błędu. Im niższa wartość MAE, tym lepiej model przewiduje rzeczywiste dane.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}. \tag{2}$$

6.4 Jak to zrobię?

Moje podejście opiera się na wybraniu modeli regresji liniowej, drzewa decyzyjnego oraz lasu losowego, które będą tuningowane za pomocą `GridSearchCV` w celu znalezienia najlepszych hiperparametrów. Wyniki są dostępne w folderze `../analysis/models_tuning.csv`. Kolejnym krokiem jest przeprowadzenie uczenia modeli i wybranie najlepszego modelu na podstawie metryk. Następnie przewidzę zarobki dla kilku tech-stacków, a na końcu przedstawię wyniki w postaci wizualizacji.

Streszczenie

W następnych rodziałach skupię się na wynikach modeli, a także na wizualizacji wyników, aby nie tworzyć zbyt długiego raportu nie będe analizować słabych modeli tylko skupię się na dwóch najlepszych modelach. **Uwaga:** Modele, które będą uczone będą umiały przewidywać zarobki na b2b albo na uop, dokładniej są to średnie z widełek.

Reszta danych: Wszyskie wyniki z uczenia zostaną zapisane w folderze `../analysis/plots/wyniki/` ew. można też podejrzeć plik z rozwiązaniem problemu w `../analysis/analysis.ipynb`.

Stosowane podziałki to 80:20, czyli 80% danych do uczenia, a 20% do testowania modelu oraz 60:40.

6.4.1 Wyniki dla podziału danych 80:20 dla umowy o pracę