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

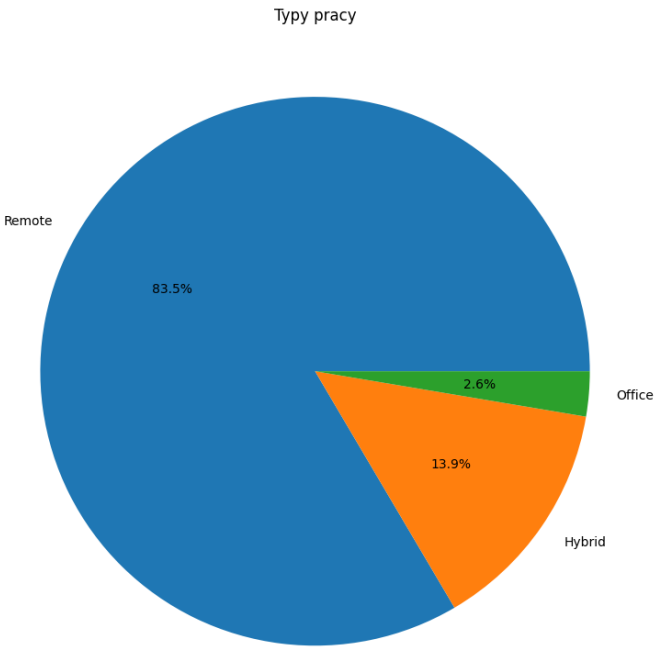
Tabela 2: Średnie zarobki w PLN dla `mida` w Polsce

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

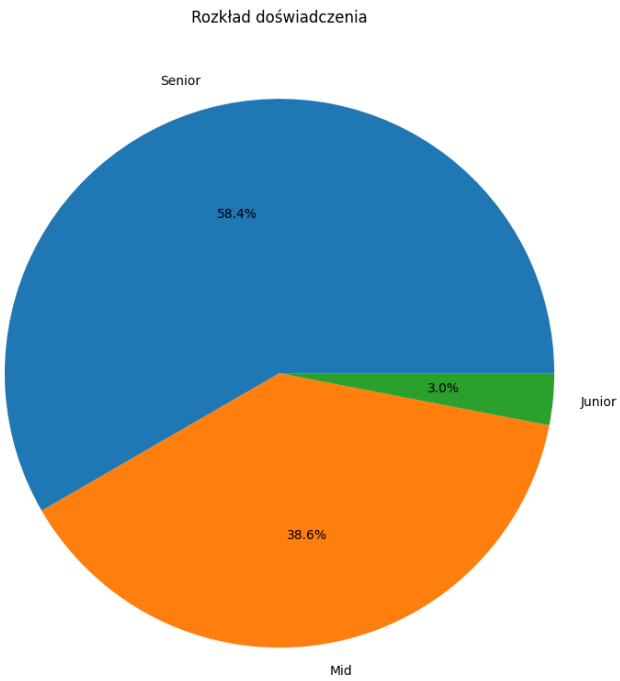
4.1 Jak się pracuje w IT?



Rysunek 1: Rozkład typów pracy

Jak widać najwięcej ofert pracy dotyczy pracy zdalnej.

4.2 Kogo szukają pracodawcy?



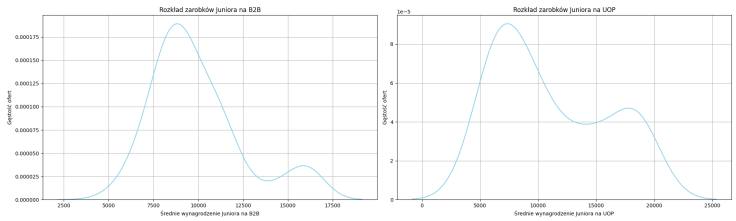
Rysunek 2: Rozkład typów pracy

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.

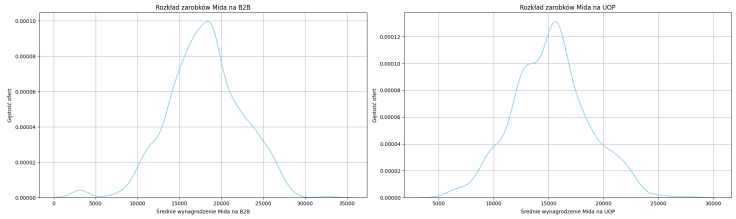
Czy to oznacza, że młodzi programiści mają trudniej, a słynne "eldorado" w IT jest tylko dla doświadczonych programistów?

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.

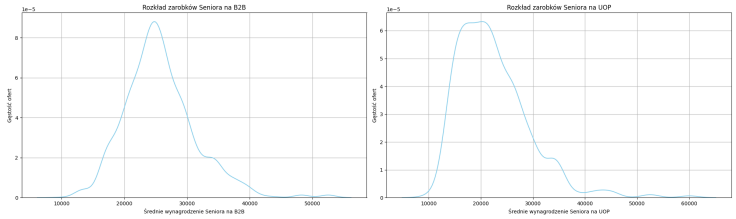
4.3 Jak rozkładają się zarobki?



Rysunek 3: Rozkłady zarobków dla poszczególnych umów dla juniorów

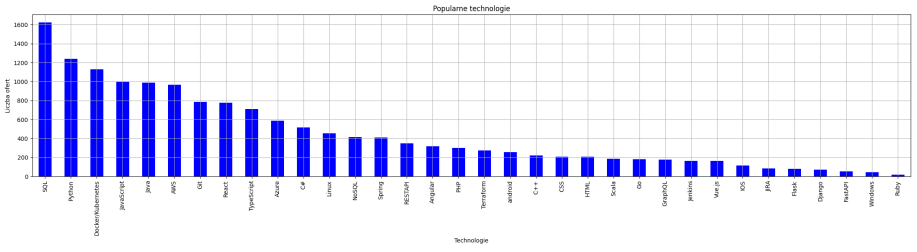


Rysunek 4: Rozkłady zarobków dla poszczególnych umów dla midów



Rysunek 5: Rozkłady zarobków dla poszczególnych umów dla seniorów

4.4 Jakie technologie są najbardziej poszukiwane?

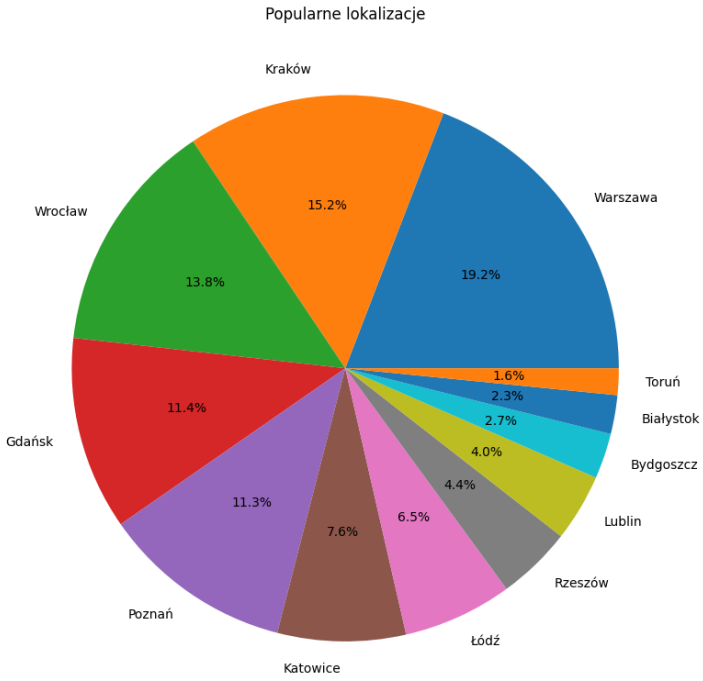


Rysunek 6: Popularne technologie w ofertach pracy w Polsce

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`.

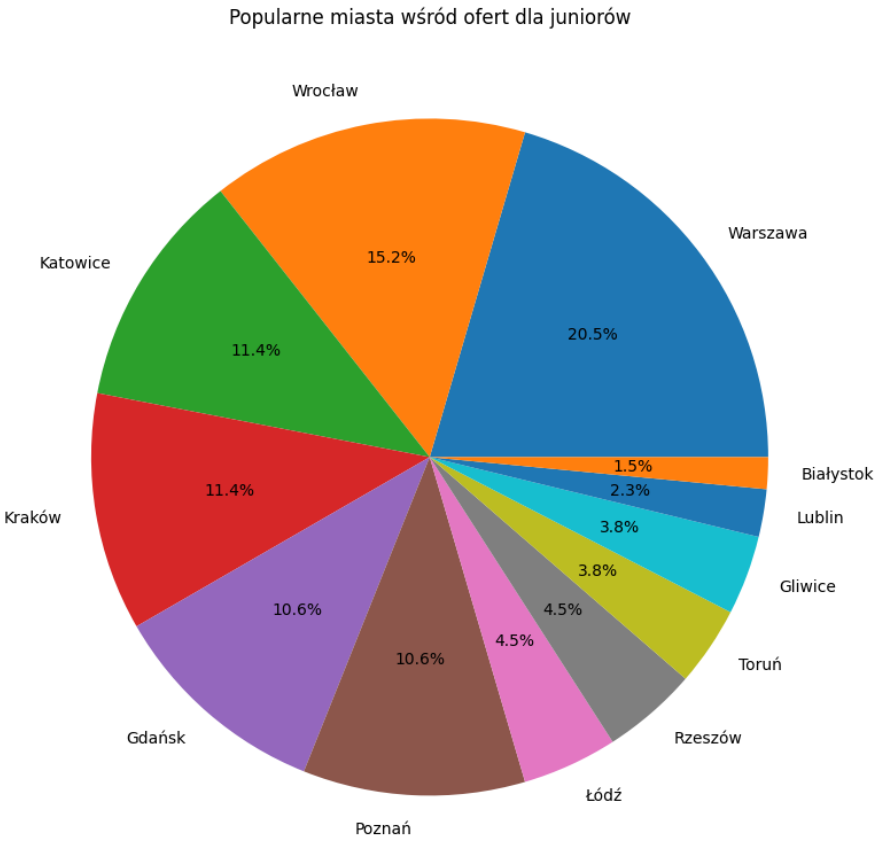
4.5 Gdzie jest największy popyt na programistów?



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?

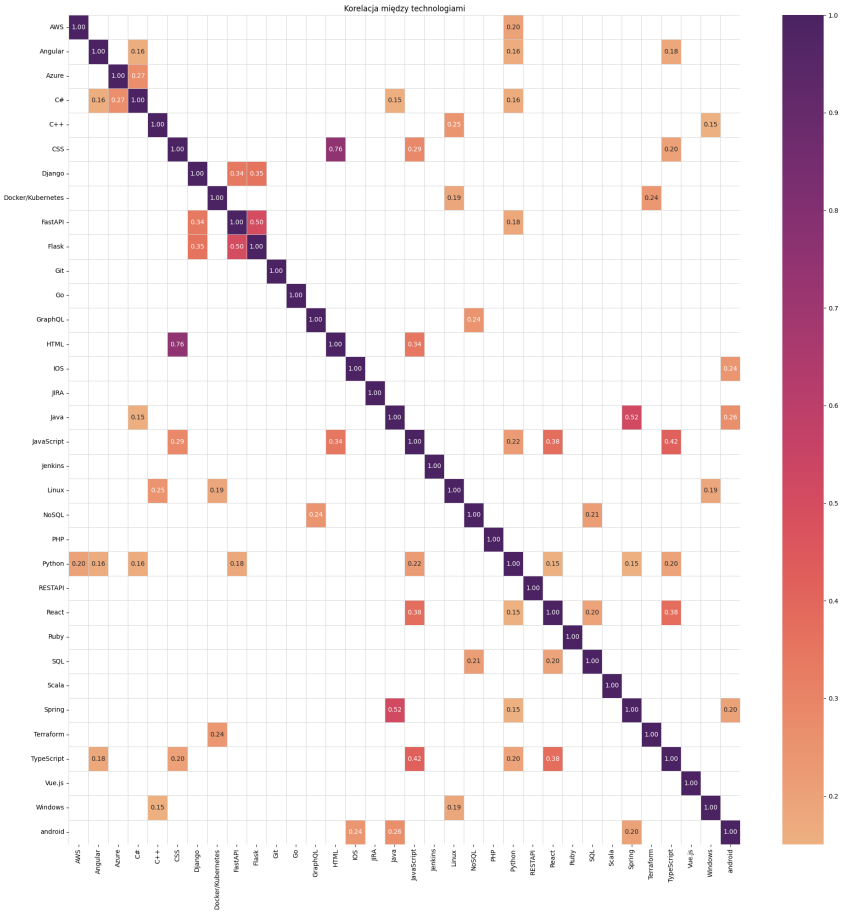


Rysunek 8: Popularne miasta w ofertach dla juniorów

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



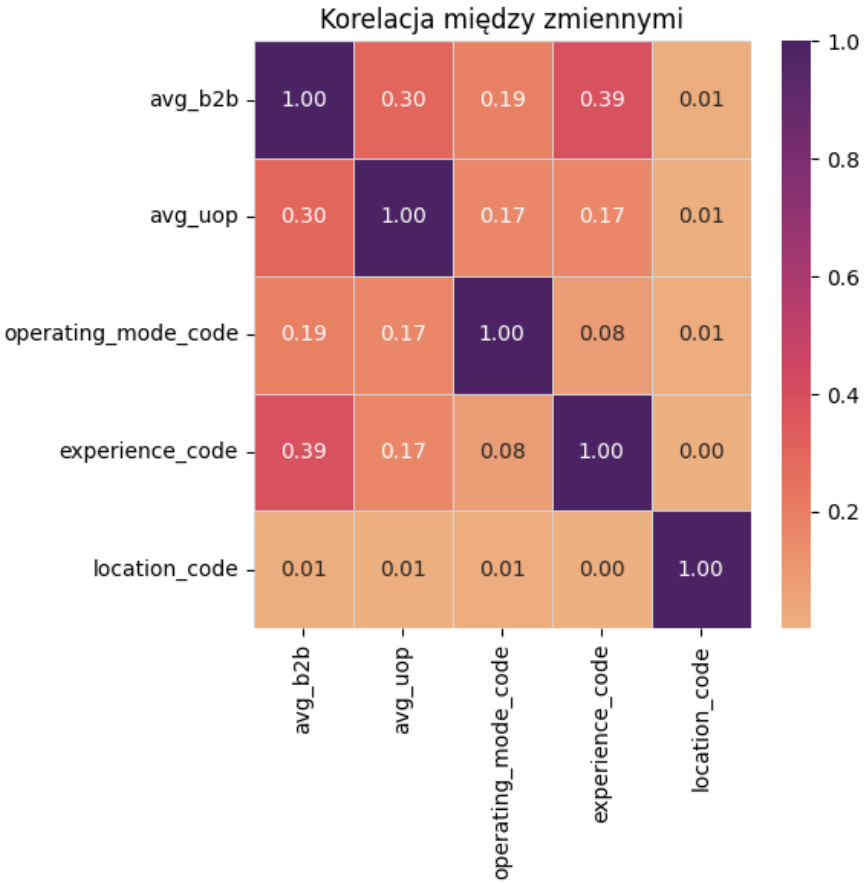
Rysunek 9: Powiązania między technologiami, zawierająca tylko wartości korelacji większe niż 0.14

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

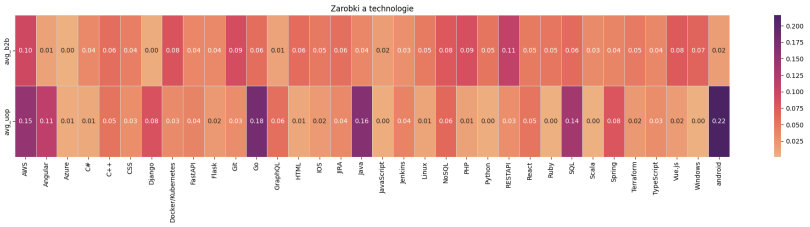


Rysunek 10: Powiązania między innymi zmiennymi

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:

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

6.3.1 Średni błąd kwadratowy

Mean Squared Error (MSE) - to metryka oceniająca jakość przewidywań modelu poprzez obliczenie średniego kwadratu różnicy między przewidywanymi a rzeczywistymi wartościami. Im niższa wartość MSE, tym lepiej model przewiduje rzeczywiste dane.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y_i}\right)^2. \tag{1}$$

6.3.2 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.

6.3.3 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.

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

6.3.4 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.

$$\text{MAE} = \frac{\sum_{i=1}^{n}|y_i - x_i|}{n}. \tag{3}$$

6.4 Wyniki oraz ograniczenia

Moje podejście opiera się na tworzeniu słownika modeli wraz z ich parametrami. Na samym początku, jeszcze przed wyborem konkretnego modelu, dokonuję doboru optymalnych parametrów dla każdego z nich. Tutaj można to zrobić na dwa sposoby, ręcznie lub za pomocą `GridSearchCV` z `sklearn`. W moim przypadku dobrałem ręcznie parametry, ponieważ po tuningu parametrów modelowi brakowało danych (?), i algorytm zwracał błąd.

Kolejnym krokiem jest przeprowadzenie procesu uczenia modelu na dostępnych danych. Po zakończeniu tego etapu prezentuję wyniki metryk dla najlepszego modelu i zwracam już wytrenowany model. Następnie udostępniam możliwość przewidywania zarobków dla większego zbioru danych.

Ograniczenia - tutaj warto zauważyć jedną rzecz, dla niektórych tech-stacków model nie jest w stanie przewidzieć zarobków, ponieważ nie ma tak dużo danych w zbiorze. W tym przypadku można bybyło rozszerzyć zbiór danych o kolejne ofert - zautomatyzować proces pobierania danych, czyli aktualizować bazę co klika dni.

Wyniki - zwracane modele mogą się różnić w zależności od danych wejściowych, czyli np. dla Java, Spring może zostać wybrany model `Ridge` a dla Pythona wybrany może być `RandomForest`, no i oczywiście zwrócony model jest już wytrenowany na danych, które zostały podane na wejściu, więc przy ponownym przewidzeniu zarobków dajemy mu tylko więcej danych. Na koniec tworze wizuazlizację wyników, gdzie mamy porównanie rzeczywistych zarobków z przewidywanymi.

Streszczenie

Proponowane tech-stacki do uczenia Na tych poszczególnych technologiach będą się uczyć modele, a następnie zostanie wybrany najlepszy

- Python
- JavaScript, React
- Docker/Kubernetes, AWS

Pozostałe dane do wyznaczenie zarobków dla powyższych tech-stacków oraz innych zmiennych:

Tech-stacki	Lokalizacja	Doświadczenie	Typ kontraktu
Python	Warszawa	Junior	B2B
JavaScript, React	Kraków	Mid	UOP
Docker/Kubernetes, AWS	Wrocław	Senior	B2B

6.4.1 Wyniki dla podziału danych 80:20

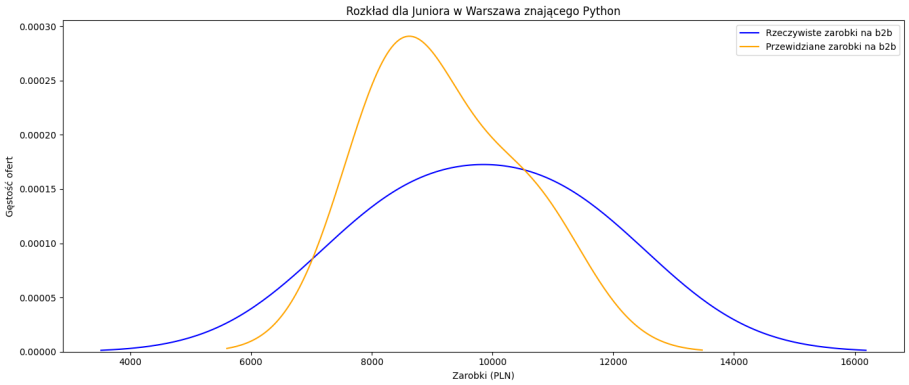
Analizę wyników przeprowadzę dla podziału danych 80:20, gdzie 80% danych to dane treningowe, a 20% to dane testowe.

Co z tego wynika?

Model będzie miał wystarczająco dużo danych do nauki w przypadku popularnych technologii, zatem dane, które będziemy przewidywać będą lepiej dopasowane w teorii.

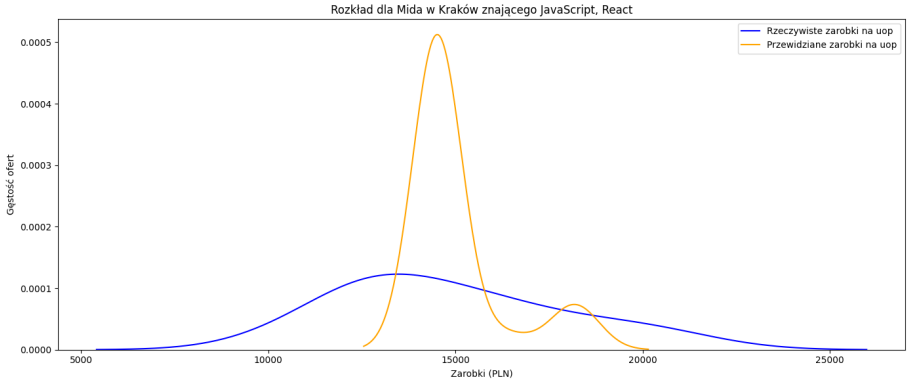
Modele	MSE	MAE	RMSE	R2
LinearRegression()	28422749.87	4358.93	5331.30	0.4238
DecisionTreeRegressor()	28263018.93	4312.16	5316.30	0.4271
RandomForestRegressor(n_estimators=20)	28284532.84	4330.01	5318.32	0.4266
Ridge(alpha=0.1)	28422322.40	2918.68	5331.26	0.4238
Lasso()	28422009.47	4358.85	5331.30	0.4238

Wybrany model to DecisionTreeRegressor(), ponieważ ma największy współczynnik determinacji R^2. Przewidywane zarobki dla Juniora znającego ['Python'] w lokalizacji Warszawa na umowie b2b to: **9400.0 PLN**.



Modele	MSE	MAE	RMSE	R2
LinearRegression()	13221438.39	2919.54	3636.13	0.4823
DecisionTreeRegressor()	21276698.63	3431.44	4612.67	0.1669
RandomForestRegressor(n_estimators=20)	18433724.47	3192.39	4293.45	0.2782
Ridge(alpha=0.1)	13213875.66	2918.68	3635.09	0.4826
Lasso()	13219274.21	2919.21	3635.83	0.4824

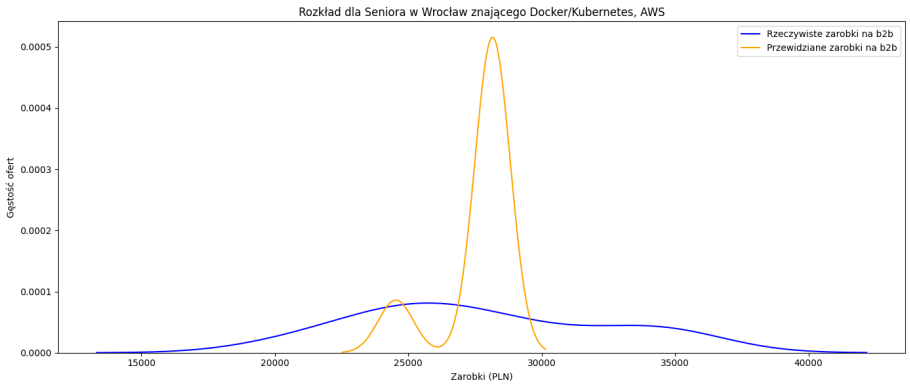
Wybrany model to `Ridge` z wartością `alpha=0.1`, ponieważ ma najniższe wartości błędów oraz najwyższy współczynnik determinacji R^2. Przewidywane zarobki dla Mida znającego ['JavaScript', 'React'] w lokalizacji Kraków na umowie uop to: **15028.69 PLN**.



Rysunek 12: Przewidywane zarobki dla mida znającego JS i React w Krakowie na uop

Modele	**MSE**	**MAE**	**RMSE**	**R2**
LinearRegression()	11084299.37	2678.04	3329.31	0.6751
DecisionTreeRegressor()	10681171.68	2605.54	3268.21	0.6870
RandomForestRegressor(n_estimators=20)	10691260.30	2643.63	3269.75	0.6867
Ridge(alpha=0.1)	11091505.44	2678.65	3330.39	0.6749
Lasso()	11085643.78	2678.07	3329.51	0.6751

Wybrany model to `DecisionTreeRegressor`, ponieważ ma najniższe wartości błędów oraz najwyższy współczynnik determinacji R^2. Przewidywane zarobki dla Seniora znającego ['Docker/Kubernetes', 'AWS'] w lokalizacji Wrocław na umowie b2b to: **27391.62 PLN**.

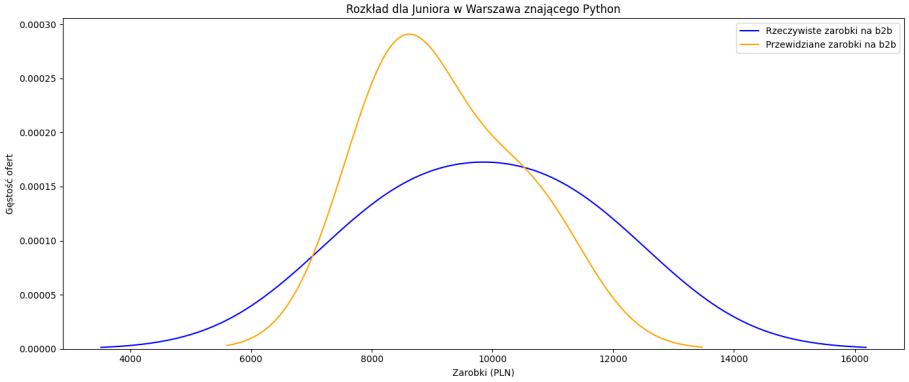


Rysunek 13: Przewidywane zarobki dla seniora w Wrocławiu znającego Docker, Kubernetes oraz AWS na b2b

6.4.2 Wyniki dla podziału danych 60:40

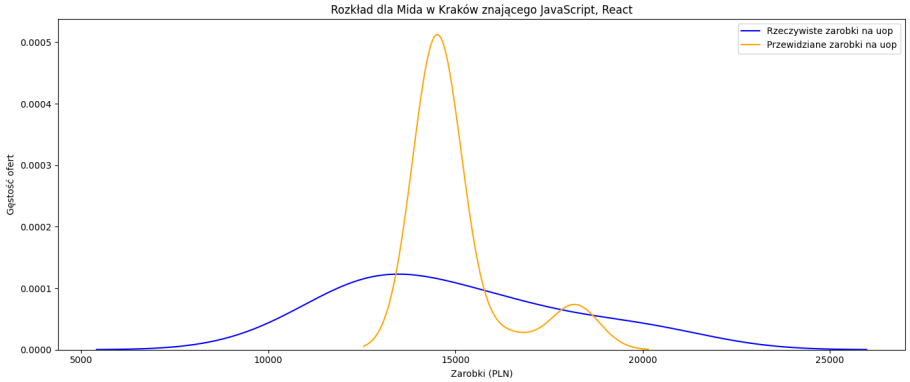
Modele	MSE	MAE	RMSE	R2
LinearRegression()	29050483.53	4406.48	5389.85	0.4097
DecisionTreeRegressor()	29520117.74	4434.43	5433.24	0.4001
RandomForestRegressor(n_estimators=20)	29210648.93	4417.47	5404.67	0.4064
Ridge(alpha=0.1)	29050015.21	4406.39	5389.81	0.4097
Lasso()	29049387.82	4406.36	5389.75	0.4238

Wybrany model to `Lasso`, ponieważ ma największy współczynnik determinacji R^2. Przewidywane zarobki dla Juniora znającego ['Python'] w lokalizacji Warszawa na umowie b2b to: **9398.07 PLN**.



Modele	MSE	MAE	RMSE	R2
LinearRegression()	17512382.62	3014.49	4184.78	0.4212
DecisionTreeRegressor()	21206971.44	3442.39	4605.10	0.2991
RandomForestRegressor(n_estimators=20)	18610952.15	3251.63	4314.04	0.3849
Ridge(alpha=0.1)	17503667.86	3014.10	4183.74	0.4215
Lasso()	17512097.61	3014.5	4184.74	0.4212

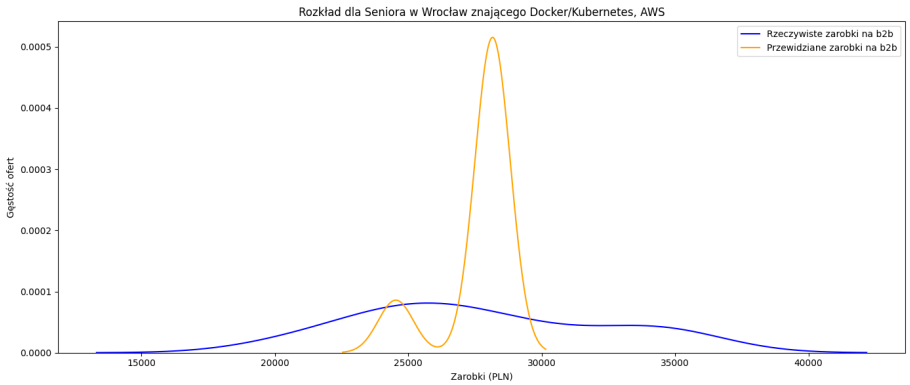
Wybrany model to `Ridge` z wartością `alpha=0.1`, ponieważ ma najniższe wartości błędów oraz najwyższy współczynnik determinacji R^2. Przewidywane zarobki dla Mida znającego ['JavaScript', 'React'] w lokalizacji Kraków na umowie uop to: **14872.58 PLN**.



Rysunek 14: Przewidywane zarobki dla mida znającego JS w React w Krakowie na uop

Modele	**MSE**	**MAE**	**RMSE**	**R2**
LinearRegression()	10035694.55	2466.69	3167.92	0.70586
DecisionTreeRegressor()	10593012.16	2564.23	3254.69	0.68953
RandomForestRegressor(n_estimators=20)	10042038.81	2513.25	3168.917	0.70568
Ridge(alpha=0.1)	10036240.1	2467.61	3168.002	0.70585
Lasso()	10036562.26	2466.89	3168.053	0.70584

Wybrany model to `LinearRegression`, ponieważ ma najniższe wartości błędów oraz najwyższy współczynnik determinacji R^2. Przewidywane zarobki dla Seniora znającego ['Docker/Kubernetes', 'AWS'] w lokalizacji Wrocław na umowie b2b to: **28221.87 PLN.**



Rysunek 15: Przewidywane zarobki dla seniora w Wrocławiu znającego Docker, Kubernetes oraz AWS na b2b

6.5 Wnioski

Co do podziału 80/20 oraz 60/40 to wyniki są bardzo podobne, ale warto zauważyć, że dla podziału 60/40 zarobki są niższe - bardziej realne. Warto zauważyć, najlepsze modele zależą od teck-stacku, ale najczęściej wyberany był model `Ridge`. Klasyczny model regresji liniowej pojawił się raz gdy mieliśmy określoną jedną technologię i była ona bardzo częsta w ofertach, czyli musi dla bardziej rozwiniętej bazy prawdopodobnie ten model byłby całkiem dobry. Warto dodać błędy `MAE` oraz `RMSE` są na poziomie kilku tysięcy PLN co jest dobrym wynikiem dla ofert skierowanych dla midów, a zwłaszcza dla seniorów, gorzej jest z juniorami, ponieważ błędy są duże jak na przewidzianą pensję, wynika to z tego, że obecnie na rynku jest relatywnie mało ofert dla tej grupy. Ostatnią rzeczą może być to, że współczynnik determinacji R^2 jest ogólnie całkiem wysoki, chociaż on wynika z tego, że ogólnie danych treningowych było sporo, gorzej może być w przypadku gdybyśmy chcieli przewidzieć niszowe technologie np. Ruby z C++. Ogónie można powiedzieć, że wyniki są trochę zawyżone co może wynikać z tego, że w ofertach pojawiały się bardzo duże zarobki, które mogą być wyjątkiem, a nie regułą.

Koniec końców, nie udało się wyuczyć jednego modelu, ale udało się stworzyć algorytm dobierający odpowiedni model do danych wejściowych. Warto dodać, że projekt może być udoskonalony o algorytm, który będzie automatyzować proces pobierania danych, czyli aktualizować bazę co klika dni, mogło by to wzbogacić bazę danych o nowe oferty pracy, a co za tym idzie poprawić wyniki modeli.