Zastosowanie wybranych algorytmów uczenia maszynowego do symulacji zmian wartości współczynnika Beta

Patrycja Lewczuk[1] Martyna Mech[2]

[1] patlew@student.agh.edu.pl

[2] mech@student.agh.edu.pl

2023

Streszczenie

Celem projektu jest zbadanie możliwości zastosowania uczenia maszynowego do symulacji zmian wartości współczynnika Beta, który pozwala zmierzyć związek między dochodowością akcji danego przedsiębiorstwa a przeciętną stopą zwrotu z wszystkich akcji. W pierwszej części niniejszej pracy skupimy się na zdefiniowaniu problemu i dostarczeniu potrzebnych definicji i kontekstu dla dalszych badań. Następnie zbudujemy model uczenia maszynowego i zbadamy uzyskane wyniki na podstawie danych historycznych i wiedzy teoretycznej.

Słowa klucze: uczenie maszynowe, współczynnik Beta, Capital Asset Pricing Model, algorytmy, ryzyko systematyczne

1 Wstęp i przegląd aktualnego stanu badań

Analiza finansowa wykorzystująca dane historyczne dotyczące akcji może ujawnić interesujące informacje, wzorce, a także może być wykorzystana jako przewodnik przy konfiguracji portfela.

Tworząc portfel, zwykle mamy tendencję do definiowania dwóch ważnych zmiennych, tj. zazwyczaj chcemy dywersyfikować i próbować kontrolować stosunek ryzyka do zwrotu.

Skupimy się na drugim problemie i przedstawimy, w jaki sposób ktoś może zastosować i wykorzystać model wyceny aktywów kapitałowych (CAPM) do oceny zmiennej stosunek ryzyka do zysku.

1.1 Uczenie maszynowe w wycenie aktywów

Uczenie maszynowe stało się w ostatnich latach obiecującym narzędziem do ulepszania modeli związanych z wyceną aktywów. Algorytmy i modele uczenia maszynowego radzą sobie z nieliniowymi relacjami i złożonymi strukturami danych, z tego względu doskonale nadają się do analizowania dużych i złożonych zestawów danych finansowych. Korzystając z uczenia maszynowego, analitycy finansowi mogą opracowywać solidniejsze i dokładniejsze modele wyceny aktywów, które uwzględniają szerszy zakres czynników, w tym dane makroeko-

nomiczne, fundamenty firmy, a nawet charakter wiadomości i danych z mediów społecznościowych. Wszystko po to, aby opracować modele, które mogą dokładnie wycenić aktywa.

Korzystanie z algorytmów uczenia maszynowego prowadzi do dokładniejszych wycen aktywów i podejmowania bardziej świadomych decyzji inwestycyjnych, ze względu na umiejętność identyfikowania wzorców i relacji w dużych, złożonych zestawach danych, których nie dostrzegają tradycyjne modele wyceny aktywów. Do tego potrafią w dość krótkim czasie przetwarzać duże ilości danych, przy czym identyfikować potencjalne zagrożenia i przewidywać trendy rynkowe, co pomaga instytucjom finansowym lepiej zarządzać ryzykiem. Dodatkowo algorytmy uczenia maszynowego mogą obsługiwać złożone struktury danych, między innymi nieustrukturyzowane dane tekstowe, co umożliwia analitykom finansowym włączenie do swoich modeli szerszego zakresu źródeł danych.

Dzięki temu uczenie maszynowe szybko staje się niezbędnym narzędziem dla inwestorów i instytucji finansowych chcących uzyskać przewagę konkurencyjną na rynku oraz lepiej zarządzać ryzykiem poprzez identyfikację potencjalnych zagrożeń i przewidywanie trendów rynkowych.

1.2 Przegląd literatury

W ostatnich latach mogliśmy obserwować dynamiczny rozwój badań nad możliwościami uczenia maszynowego w sektorze finansowym. Jest to temat chętnie eksplorowany przez banki i instytucje finansowe, ale także środowisko naukowe. Przytoczymy teraz kilka wybranych publikacji, które zarysują obraz tego, jak przedstawiają się dotychczasowe osiągnięcia oraz zidentyfikowane problemy.

Jednym z tematów cieszących się największą popularnością są projekty dotyczące predykcji cen aktywów w różnych modelach finansowych. Przykładem są badania poczynione w artykule "Machine Learning Algorithms for Financial Asset Price Forecasting"[4]. Autor ocenia, że algorytmy uczenia maszynowego są skuteczne w prognozowaniu cen aktywów finansowych i mogą prowadzić do bardziej dokładnych prognoz w porównaniu z tradycyjnymi metodami. Podkreśla przy tym jeden ważny problem: wrażliwość modeli uczenia maszynowego na błędy w danych, które mogą prowadzić do niewłaściwych prognoz. To ważna uwaga, którą należy odpowiednio zaadresować również w niniejszej pracy.

W artykule "Estimating Security Betas via Machine Learning"[1], porównano dokładność predykcji współczynnika beta za pomocą uczenia maszynowego i pewnych tradycyjnych matematycznie wyznaczonych estymatorów. Autorzy uzyskali wnioski o przewadze podejścia machine learningowego z wyróżnieniem algorytmu Random Forest. Jest to obiecujący wynik, z który będziemy mogli wykorzystać w dalszej części pracy.

Pozytywne konkluzje o dokładności algorytmów uczenia maszynowego mają także autorzy "Machine Learning Methods in Finance: Recent Applications and Prospects"[2], lecz wymieniają także pewne wady. Ich zdaniem metody machine learningu nie zapewniają odpowiedniego wytłumaczenia dla swoich wyników i choć zwracane błędy predykcji przyjmują niskie wartości, nie zawsze łatwo zaobserwować, w jaki sposób algorytm wyprodukował dany wynik. Inną niedogodnością, jest potrzeba posiadania bardzo duży zbiorów danych, jeżeli chcemy uzyskać wirygodne i dobre predykcje. Ostatnim problemem, który zostaje pokreślony są wysokie koszty obliczeniowe w porównaniu z tradycyjnymi metodami, takimi jak regresja liniowa.

1.3 Model wyceny aktywów kapitałowych (CAPM)

W branży finansowej od długiego czasu do szacowania wartości aktywów wykorzystuję się tradycyjne modele wyceny aktywów. Jednym z najpopularniejszych modelów jest model wyceny aktywów kapitałowych (ang. The Capital Asset Pricing Model; w skrócie CAPM), który za pomocą regresji liniowej, opisuje związek między oczekiwanym zwrotem z aktywów a systematycznym ryzykiem rynkowym w celu oszacowania oczekiwanego zwrotu z aktywów. Oblicza oczekiwane zwroty z inwestycji i może służyć do ustalania cen poszczególnych papierów wartościowych, takich jak akcje.

Głównymi założeniami modelu CAPM są:

- liniowe przedstawienie zależności między stopą zwrotu z inwestycji a systematycznym ryzykiem rynkowym
- jedyny wskaźnik charakteryzujący wyżej wspomnianą zależność to współczynnik Beta (zwany również indeksem Beta), któremu zakłada się stałość w czasie
- zależności dotyczą tylko jednego okresu

CAPM wskazuje, że oczekiwany zwrot z aktywów jest równy zwrotowi wolnemu od ryzyka powiększonemu o premię za ryzyko. Zakłada, że inwestorzy działają racjonalnie i chcą maksymalizować zwrot oraz ograniczać ryzyko jak najbardziej jest to możliwe. Zatem celem CAPM jest obliczenie, jakiego zwrotu inwestor może się spodziewać przy danej premii za ryzyko w stosunku do stopy wolnej od ryzyka.

Matematycznie formułę CAPM może zapisać w następujący sposób:

$$R_i = R_f + \beta_i \cdot (R_m - R_f),$$

gdzie:

R_i – oczekiwana stopa zwrotu z aktywów, które szacuje model

R_f – stopa wolna od ryzyka i odnosi się do teoretycznej stopy zwrotu z inwestycji przy zerowym ryzyku

β_i – współczynnik Beta (współczynnik ryzyka rynkowego firmy)

R_m – średnia rynkowa stopa zwrotu z akcji; ogólny zwrot z rynku

$\beta_i \cdot (R_m - R_f)$ – premia za ryzyko.

Mimo wszystko CAPM nie jest idealnym rozwiązaniem. Ze względu na częste zmiany stopy wolnej od ryzyka i zbyt wiele założeń, model CAPM może nie być dokładny. Dodatkowo jest uzależniony od ryzyka systematycznego i nie bierze pod uwagę innych czynników np. jak względna wrażliwość na inflację.

1.4 Współczynnik Beta

Jednym z istniejących w finansach podziałów ryzyka jest rozróżnienie na ryzyko systematyczne i niesystematyczne (specyficzne). Ryzyko systematyczne to inaczej ryzyko rynkowe, biorące pod uwagę kompletny przekrój rynku. Jest związane z sytuacją makroekonomiczną i inwestorzy nie mają na nie wpływu. Ryzyko niesystematyczne to natomiast ryzyko, którym

obarczona jest inwestycja w akcje konkretnej spółki.

Współczynnik Beta jest miarą ryzyka systematycznego. Reprezentuje prawdopodobieństwo tego, jak cena akcji danej spółki zmieni się, biorąc pod uwagę zmiany, którym uległa stopa zwrotu rynku. Często badanym w literaturze przykładem jest wyznaczanie bety dla funduszu indeksowego S&P500 i akcji firm, które się w nim znajdują, np. Apple.

Współczynnik beta przedsiębiorstwa i wyznaczany jest ze wzoru:

$$\beta_i = r_{i,m} \cdot \frac{\sigma_i}{\sigma_m} \tag{1}$$

lub równoważnie:

$$\beta_i = \frac{\mathrm{cov}_{i,m}}{\sigma_m^2}, \tag{2}$$

gdzie

$r_{i,m}$ – współczynnik korelacji występującej między rentownością akcji spółki i a rynkiem m,

σ_i – odchylenie standardowe zwrotów z akcji przedsiębiorstwa i,

σ_m – odchylenie standardowe zwrotów z indeksu rynkowego m,

$\mathrm{cov}_{i,m}$ – kowariancja rentowności akcji przedsiębiorstwa i oraz rynku m,

σ_m^2 – wariancja rentowności rynku m.

Interpretacja wartości współczynnika beta jest dość prosta. Omówmy wszystkie możliwe przypadki i ich znaczenie dla inwestycji w akcje danego przedsiębiorstwa i.

- Jeżeli $\beta_i < 0$, to akcje firmy są skorelowanie ujemnie ze stopą zwrotu rynku. Obserwujemy wtedy, że ceny akcji przedsiębiorstwa spadają, podczas gdy stopa zwrotu rynku rośnie i odwrotnie - ceny akcji przedsiębiorstwa rosną, gdy stopa zwrotu rynku maleje. Na przykład, gdyby stopa zwrotu z indeksu rynkowego wzrosła o 2%, a β_i wynosiła -2, to cena akcji firmy zmalałaby w tym samym czasie o 4%.

- Jeżeli $\beta_i = 0$, to stopa zwrotu z akcji jest całkowicie niezależna od stopy zwrotu rynku, ceny takich akcji nie reagują na zmiany na rynku. Betę równą 0 mają na przykład obligacje.

- Jeżeli $\beta_i \in (0,1)$, to pomiędzy stopami zwrotu z indeksu rynkowego i cen akcji przedsiębiorstwa istnieje dodatnia korelacja, jednak wzrosty/spadki cen akcji przedsiębiorstwa nie są tak duże jak wzrosty/spadki cen na rynku. Ryzyko tych akcji jest mniejsze od ryzyka rynkowego.

- Jeżeli $\beta_i = 1$, to akcje przedsiębiorstwa są silnie skorelowane z rynkiem. Znaczy to, że jeżeli wartość stopy zwrotu z indeksu rynkowego spada/wzrasta, to ceny akcji przedsiębiorstwa spadają/wzrastają w identyczny sposób.

- Jeżeli $\beta_i > 1$, to zmienność ceny akcji przedsiębiorstwa ma ten sam kierunek co zmienność wartości rynku, ale wzrosty lub spadki są bardziej znaczące, a co za tym idzie, inwestycja jest obarczona większym ryzykiem. Załóżmy na przykład, że stopa zwrotu z indeksu rynkowego wzrosłaby o 2%, zaś β_i wynosiła 2. To byłoby równoważne, z tym że cena akcji przedsiębiorstwa wzrosłaby w tym samym czasie o 4%.

2 Sformułowanie problemu i proponowane rozwiązanie

Algorytmy uczenia maszynowego są wykorzystywane w wycenie aktywów do analizowania dużych ilości danych finansowych, identyfikowania wzorców i związków oraz tworzenia dokładniejszych i niezawodnych modeli wyceny aktywów. Analitycy finansowi wykorzystują algorytmy uczenia maszynowego do analizowania różnych źródeł danych, w tym danych makroekonomicznych, fundamentów firm, nastrojów w wiadomościach oraz danych z mediów społecznościowych, aby tworzyć modele, które dokładnie wyceniają aktywa.

2.1 Opis problemu do rozwiązania

Celem niniejszej pracy jest zbadanie zastosowania wybranych algorytmów uczenia maszynowego w symulacji zmian wartości współczynnika Beta dla aktywów finansowych. Jak zauważyliśmy w części pierwszej na podstawie dostępnej literatury, tradycyjne metody szacowania Beta mogą być podatne na różne ograniczenia i założenia, które mogą wpływać na dokładność wyników. Algorytmy uczenia maszynowego oferują alternatywne podejście, które daje obiecujące wyniki pod względem zwiększenia elastyczności i dokładności prognozowania Bety.

Pierwszym krokiem w analizie problemu będzie pobranie odpowiednich danych finansowych. Na tym etapie badań, kluczowe jest upewnienie się, że posiadamy pełne i wiarygodne dane w zakresie czasowym, który poddajemy badaniom. Szczęśliwie, dane finanasowe spółek są jawnie dostępne na wielu platformach jak Yahoo Finanace czy STOOQ. Do obliczenia współczynnika Beta potrzebna jest informacja o rentownosci rynku oraz pojedynczej spółki. W naszym przypadku za cały rynek uznamy indeks Standard & Poor's 500 (S&P500). Jest on jednym z najbardziej znanych i reprezentatywnych wskaźników giełdowych na świecie. To indeks wagowy, który obejmuje 500 największych i najbardziej płynnych spółek notowanych na giełdzie amerykańskiej. W jego skład wchodzą akcje spółek takich jak Apple, Microsoft, Amazon, Google i Facebook. Zestawimy dane o cenach akcji tych spółek z danymi o wartości indeksu S&P500 na przestrzeni lat i na tej podstawie będziemy w stanie policzyć historyczne wartości Bety.

Interesującym problemem będzie również dobór częstotliwości i zakresu danych, których poddamy modelowaniu. Temu zagadnieniu przyjrzeli się też autorzy [3], próbując predykcji na różnych horyzontach czasowych, np. jednego miesiąca, dwunastu miesięcy, sześćdziesięciu miesięcy. Posiadamy dostęp do danych sięgających kilkanaście lat wstecz, toteż mamy szansę na podobnie szeroki czasowo eksperyment.

Na podstawie wiedzy teoretycznej i innych przesłanek, powinniśmy wybrać kilka modeli uczenia maszynowego, które po treningu na danych historycznych zwrócą swoje predykcje wartości Bety. Ostatecznie, skuteczność modeli trzeba będzie osądzić na podstawie wybranych miar. Popularnymi metrykami, które sprawdzą się również w naszym eksperymencie są:

- Błąd średniokwadratowy (MSE) – mierzy średnią kwadratową różnicę między wartościami przewidywanymi przez model, a rzeczywistymi wartościami Beta. Im niższa wartość MSE, tym lepsza jakość modelu. MSE ma jednak tendencję do bardziej penalizowania dużej wartości błędów, co oznacza, że silne odstępstwa mogą znacząco wpływać na wynik. Alternatywnie, użyc można maiary RMSE, która jest pierwiastkiem kwadratowym MSE. Jej zaletą jest to, że jej jednostka pokrywa się z jednostką danych.

- Współczynnik determinacji (R^2) – mierzy, jak dobrze model dopasowuje się do danych, wskazując odsetek wariancji wartości Beta, który jest wyjaśniany przez model. Wartość

R^2 w zakresie od 0 do 1, gdzie 1 oznacza idealne dopasowanie. R^2 wskazuje, jak wiele zmienności wartości Beta jest objaśniane przez zmienne w modelu.

- Błąd bezwzględny (MAE) – mierzy średnią bezwzględną różnicę między wartościami przewidywanymi przez model, a rzeczywistymi wartościami Beta. Jest to miara oparta na wartościach bezwzględnych, więc nie bierze pod uwagę kierunku błędów. Im niższa wartość MAE, tym lepsza jakość modelu.

2.2 Sposoby zdobywania wiedzy przez systemy uczenia maszynowego

Uczenie maszynowe to metoda pozwalająca systemom komputerowym na zdobywanie wiedzy poprzez analizę zgromadzonych danych. Metody uczenia maszynowego są podzielone na następujące rodzaje:

- Uczenie nadzorowane – stosowane w wycenie aktywów w celu przewidywania ich wartości na podstawie danych historycznych. Te algorytmy korzystają z oznakowanych danych, aby nauczyć się wzorców i relacji między zmiennymi, a następnie wykorzystują to uczenie do przewidywania przyszłych wartości aktywów.

- Uczenie nienadzorowane – stosowane w wycenie aktywów w celu analizowania dużych, złożonych zbiorów danych i identyfikowania wzorców i relacji, które mogą być trudne do zidentyfikowania dla ludzkich analityków. Te algorytmy nie polegają na danych oznakowanych i mogą odkrywać wcześniej nieznane wzorce w danych.

- Uczenie przez wzmacnianie – wykorzystywane w wycenie aktywów do optymalizacji strategii inwestycyjnych poprzez naukę na podstawie danych historycznych i dostosowywanie odpowiednich decyzji inwestycyjnych. Te algorytmy mogą identyfikować optymalne strategie inwestycyjne na podstawie wcześniejszej wydajności i warunków rynkowych.

2.3 Wybrane metody uczenia maszynowego do predykcji

Po przeanalizowaniu różnych metod pozyskiwania wiedzy przez uczenie maszynowe, czas wybrać najlepszą technikę, która zautomatyzuje proces generowania przewidywań przyszłych wartości współczynnika Beta. Ze względu na posiadanie dostępu do wszystkich potrzebnych historycznych danych (które mogą również stanowić dane wyjściowe), postanowiono zrezygnować z dalszych prac opartych na uczeniu nienadzorowanym. Dodatkowo celem projektu jest symulacja zmian wartości współczynnika Beta, a nie tworzenie autonomicznego systemu spekulującego bez ingerencji człowieka. W związku z tym odrzucono podejście uczenia przez wzmacnianie. Ostatecznie, najlepszą metodą nauki systemu okazało się podejście nadzorowane, spełniające wcześniej zdefiniowane założenia projektu.
Najpopularniejszymi metodami uczenia maszynowego wspierające podejście nadzorowane są między innymi:

- Regresja liniowa – metoda pozwalająca na wyuczenie modelu reprezentującego liniową zależność między danymi wejściowymi a wyjściowymi.
  Podczas generowania predykcji zmiennych objaśnianych, zakłada się następujące założenia: niezależność i addytywność (każdy czynnik wpływający na proces doboru parametrów i predykcji wyników uznawany jest za niezależny i dodawany jest do pozostałych podczas obliczania wartości zmiennej przewidywanej) oraz monotoniczność i liniowość

(zmiana wartości zmiennych objaśniających nie prowadzi do utraty liniowości i zmiany kierunku prostej odwzorowującej generowane predykcje).
Regresja liniowa prognozuje wartości, wykorzystując ważoną sumę cech danych wejściowych oraz wyraz wolny zgodnie z poniższym wzorem:

$$y = b_0 + b_1 x_1 + b_2 x_2 + ... + b_n x_n, \tag{3}$$

gdzie

- y – zmienna objaśniana,
- n – ilośc cech,
- x_i – wartość i-tej cechy,
- b_j – wartość j-tego współczynnika regresji z uwzględnieniem wyrazu wolnego.

- Algorytm k najbliższych sąsiadów - metoda, która nie wymaga trenowania modelu, aby generować predykcje zmiennych objaśnianych. Jego idea polega na przyporządkowaniu każdemu zestawowi cech danych wejściowych i umieszczeniu ich w wielowymiarowej przestrzeni na podstawie miary podobieństwa. W przypadku przekazania próbki nieoznaczonej do algorytmu wyszukiwane są k najbliższe obiekty za pomocą określonej metody. Do tego celu najczęściej stosowane są następujące miary odległości:
  - Euklidesowa
  $$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$
  - Manhattan/Taksówkowa
  $$d = \sum_{i=1}^{n} |x_i - y_i|$$
  - Minkowskiego
  $$d = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

  gdzie

  - x_i i y_i – obserwacje,
  - p – parametr (dla Euklidesowej równa się 2, a dla taksówkowej 1).

  W kolejnym etapie algorytmu k-najbliższych sąsiadów dokonuje się zliczenia liczby wystąpień każdej klasy ze zbioru k najbliższych sąsiadów i przyporządkowuje się etykietę najczęściej występującej klasy do zmiennej objaśnianej. W ten sposób dokonuje się klasyfikacji nowej próbki na podstawie etykiet najbliższych sąsiadów, które zostały już wcześniej przypisane do klas.

- Drzewa decyzyjne i lasy losowe – metoda generowania przyszłych predykcji na podstawie zmiennych objaśniających, polegająca na wykorzystaniu szeregu zasad decyzyjnych wytrenowanych na zbiorze uczącym.
  Jednym z problemów w efektywnym wykorzystaniu tej metody jest dobór odpowiedniej struktury drzewa decyzyjnego dla danego zagadnienia. W tym celu stosuje się algorytmy rekurencyjne, które umożliwiają maksymalizację zdobywania najważniejszych informacji w procesie podejmowania decyzji oraz podziału w każdym węźle drzewa.

- Sieci neuronowe – idea stojąca za nimi polega na naśladowaniu pracy ludzkiego mózgu, lub jak on działa.
  Perceptron jest jedną z najprostszych struktur w dziedzinie sieci neuronowych. Opiera się na sztucznym neuronie zwanym TLU, którego wejście i wyjście może być tylko liczbami. TLU oblicza ważoną sumę swoich cech wejściowych:

$$z = (w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n),$$

  a następnie wykorzystuje ją w funkcji tzw. kroku i generuje wynik.
  Najprostszym rodzajem sieci neuronowej działającej w jednym kierunku jest tzw. "Single Layer Perceptron", który składa się tylko z jednej warstwy wyjściowej. Wejścia są wstawiane bezpośrednio do wyjść poprzez serię wag. Sieć oblicza liniową kombinację między wagami a wejściami. Ten typ sieci neuronowych może być używany tylko do binarnej klasyfikacji liniowej. Kiedy wyniki kombinacji liniowej przekroczą próg, wyjście będzie pozytywną klasą, w przeciwnym razie wynik będzie ujemny.
  Bardziej złożonym algorytmem działającym w jednym kierunku jest "Multi-Layer Perceptron"(MLP). Składa się on z warstwy wejściowej, jednej lub więcej warstw TLU zwanych warstwami ukrytymi oraz ostatniej warstwy TLU zwanej warstwą wyjściową. Każda warstwa jest w pełni połączona z następną warstwą, a wszystkie warstwy mają neuron obciążenia oprócz warstwy wyjściowej.
  Istnieją również tak zwane rekurencyjne sieci neuronowe, które pozwalają na przesyłanie sygnałów wyjściowych z dalszych lub bliższych warstw sieci neuronowej do neuronów w warstwie wejściowej, lub w ukrytych warstwach. Dzięki temu sieci te umożliwiają wykonywanie bardziej złożonych obliczeń i posiadają bogatsze właściwości niż sieci, które przesyłają informacje jedynie w jednym kierunku (wcześniej wspomniane).

2.4 Wykorzystywane biblioteki

Eksperymenty przeprowadzimy w języku Python z użyciem jego wbudowanych bibliotek, zawierających funkcje zdolne do pobrania danych finansowych, implementowania modeli uczenia maszynowego i miar oceny tych modeli.

1. yfinance

   Działa jako rozszerzenie dla usługi Yahoo Finance API, umożliwiając łatwe i wygodne pobieranie danych finansowych do analizy, takich jak notowania giełdowe, historyczne ceny akcji, dane dotyczące indeksów, walut i towarów.

2. matplotlib

   Biblioteka służąca do generowania wykresów i wizualizacji danych. Umożliwia tworzenia różnych typów wykresów, takich jak wykresy liniowe, słupkowe, histogramy, wykresy punktowe itp.

3. numpy

   To biblioteka numeryczna, która zapewnia wsparcie dla operacji na dużych tablicach i macierzach wielowymiarowych. numpy oferuje wiele funkcji matematycznych i algebraicznych, które są przydatne podczas przetwarzania danych numerycznych. Jest również wykorzystywany jako podstawa do wielu innych bibliotek, w tym pandas i scikit-learn.

4. pandas

   To potężna biblioteka do manipulacji i analizy danych. Oferuje łatwy w użyciu interfejs do wczytywania, przetwarzania, filtrowania i agregowania danych. pandas umożliwia manipulację i analizę danych w formie tabelarycznej, zwanej DataFrames. Jest szeroko stosowana w analizie danych, eksploracji danych i przygotowywaniu danych do modelowania uczenia maszynowego.

5. scikit-learn

   Jest biblioteką przeznaczoną do uczenia maszynowego. Dostarcza szeroki wybór algorytmów uczenia maszynowego, w tym metody nadzorowanego uczenia (klasyfikacja, regresja), nienadzorowanego uczenia (klastrowanie, redukcja wymiarowości) i wzmacnianego. Można w niej znaleźć popularne algorytmy, takie jak regresja liniowa, lasy losowe, maszyny wektorów nośnych (SVM), sieci neuronowe, metody grupowania k-średnich i wiele innych. Udostępnia także narzędzia do przygotowywania danych przed użyciem ich w machine learningu, np. normalizacja, skalowanie, kodowanie zmiennych kategorycznych, obsługa brakujących danych i wiele innych. Ponadto oferuje funkcje oceny modeli, takie jak walidacja, podział na zbiór treningowy i testowy, szeroką gamę metryk jak R^2 czy RMSE.

6. keras

   Jest otwartoźródłową biblioteką programistyczną do uczenia maszynowego, która umożliwia łatwe i szybkie tworzenie modeli. Dostarcza gotowe moduły ze zdefiniowanymi parametrami, które można wykorzystać do tworzenia modeli. Te moduły mogą obejmować abstrakcyjne reprezentacje funkcji aktywacji, funkcje kosztu oraz warstwy sieci neuronowej.

7. tensorflow

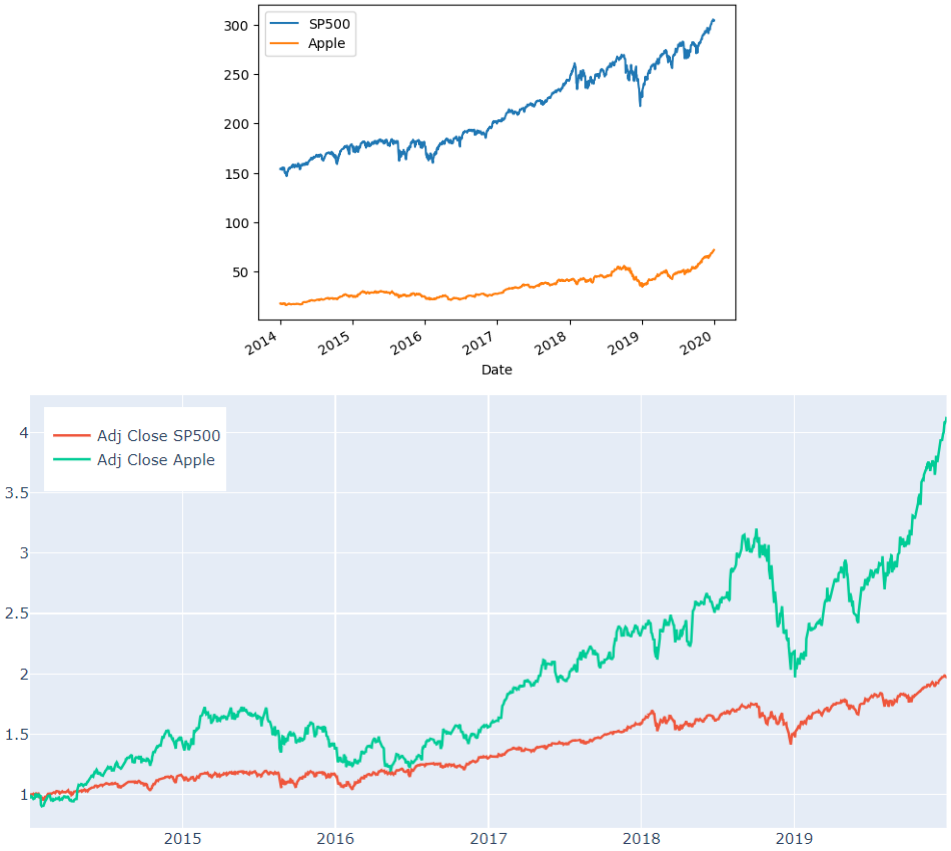
   Stworzona przez Google biblioteka, jest jednym z najważniejszych narzędzi do tworzenia, trenowania i wdrażania modeli uczenia maszynowego. Wykorzystuje grafy obliczeniowe, które pozwalają na definiowanie i manipulację operacjami matematycznymi na tensorach. To umożliwia skonstruowanie różnorodnych modeli, w tym głębokich sieci neuronowych. Oferuje bogate zestawy narzędzi i funkcji optymalizacyjnych, które pomagają w procesie tworzenia skutecznych modeli uczenia maszynowego.

3 Wyniki badań eksperymentalnych

3.1 Przygotowanie danych

Aby móc rozpocząć badanie, musimy pobrać dane historyczne akcji dla wybranych przez nas firm. Aby pobrać dane historyczne, użyłyśmy wcześniej wspomnianej biblioteki yfinance. Okres wybrany przez nas do wykorzystania w badaniach, to lata 2014-2019 dla firmy AAPL (Apple) oraz dla rynku SPY (S&P500).

Jak widać na rysunku 1, ceny akcji mają inne zakresy wartości. Aby dane były porównywalne, musimy znormalizować ceny, dzieląc je wszystkie dla każdej akcji przez jej pierwszą cenę. W ten sposób pierwszą znormalizowaną ceną będzie 1, a wartości powyżej niej będą wskazywać na wzrost ceny, natomiast wartości poniżej będą odnosić się do spadków ceny. Normalizacja pozwoli nam porównać wyniki różnych akcji.



Rysunek 1: Wartości akcji w latach 2014-2019 przed (góra) i po normalizacji (dół).

Aby móc policzyć betę, policzyłyśmy osobno dla każdego wiersza dzienny zwrot, a następnie skorzystałyśmy z następującego wzoru:

$$\beta = \frac{\text{Covariance}(R_e, R_m)}{\text{Variance}(R_m)},$$

gdzie R_e to cena akcji, R_m to dzienny zwrot.

3.2 Przeprowadzenie eksperymentów

Gotowy zbiór danych podzielono odpowiednio na zbiór treningowy i testowy według predykowanych lat. Postanowiłyśmy predykować wartości współczynnika beta dla roku 2019. Z tego powodu zgodnie z zaleceniami i informacjami umieszczonymi w artykułach przez ekonomistów, do predykcji wykorzystamy informacje z wcześniejszych 5 lat.

Na początku eksperymentów wykorzystano liniowe i drzewiaste modele uczenia maszynowego z ich domyślnymi parametrami, a następnie sieci neuronowych, tzn. MLP (jedna warstwa dla 128 neuronów z funkcją aktywacji `relu` i ostatnią warstwą Dense z 1 neuronem, kompilowany z optymalizatorem `adam` i funkcją straty `mae`) i LSTM.

LSTMy (skrót od Long Short-Time Memory) są rozszerzeniem wcześniejszych rekurencyjnych sieci neuronowych, zdolnymi do zachowania pamięci długoterminowej i wykorzystania jej do uczenia się wzorców w dłuższych sekwencjach danych źródłowych. Mogą trzymać się długoterminowych wzorców, które odkrywają podczas przechodzenia przez swoje pętle. Nasza początkowa rekurencyjna sieć neuronowa była zbudowana z jednej warstwy LSTM dla 128 neuronów, dwóch warstw Dense dla 128 neuronów i ostatnią również Dense, ale dla 1 neurona.

Model został skompilowany z optymalizatorem `adam` i funkcją straty `mae`.

Po początkowych wynikach podjęłyśmy kolejną próbę dla innych parametrów modeli. Całkowicie zrezygnowałyśmy z algorytmu regresji liniowej, gdyż wychodziła najgorzej ze wszystkich metod i nie istniały parametry, aby umożliwić jego poprawę.
Dla lasów losowych zmieniano parametry takie jak `n_estimators`, `max_depth`, `criterion` lub parametry związane z minimalną liczbą próbek.
Dla KNN zmieniano natomiast parametry takie jak rozmiar liścia i funkcji wagi oraz liczba sąsiadów i algorytm obliczający najbliższych sąsiadów i wartość zmiennej p.
Dla MLP i LSTM na początku zmieniono tylko optimizer, gdyż tylko to wystarczyło, aby błędy zmieniły się odpowiednio na lepsze.

Nie mogąc już polepszyć aktualnych modeli ML'owych i porzucając dalsze polepszanie MLP, postanowiłyśmy skupić się tylko na LSTM, dokładając różne warstwy, aby sprawdzić, czy wtedy wyniki również się polepszą.
Architektury, jakie stworzono dla LSTM to:

1. warstwa LSTM ze 128 neuronami, dwie Dense z 128 neuronami i Dense z 1 neuronem,
2. warstwa LSTM z 64 neuronami, dwie Dense z 64 neuronami i Dense z 1 neuronem,
3. dwie warstwy LSTM ze 128 neuronami, dwie Dense z 128 neuronami i Dense z 1 neuronem,
4. warstwa LSTM ze 128 neuronami, jedna Dense z 128 neuronami i Dense z 1 neuronem,
5. warstwa LSTM z 256 neuronami, dwie Dense z 128 neuronami i Dense z 1 neuronem,
6. warstwa LSTM z 256 neuronami, dwie Dense z 256 neuronami i Dense z 1 neuronem.

Wiedząc już, która architektura wychodzi najlepiej, sprawdziłyśmy, jak wygląda predykcja na rok 2019 na podstawie różnej ilości danych historycznych. Wybrałyśmy okres 5 lat, 10 lat i roku.

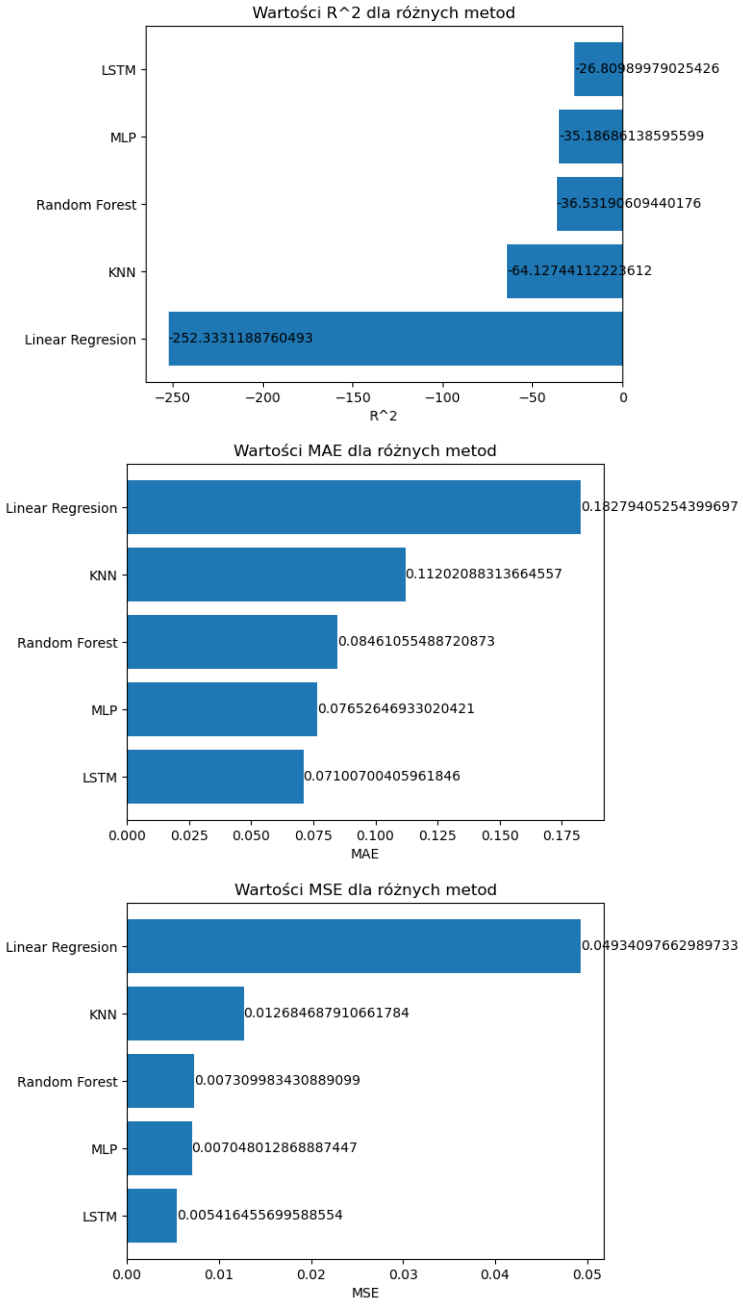
3.3 Wyniki

Dla domyślnych wartości parametrów modeli uczenia maszynowego można zauważyć na rysunku 2, że najlepiej współczynnik bety predykowały sieci neuronowe na czele z rekurencyjnymi, a najgorszy wynik wyszedł dla regresji liniowej.

Po polepszaniu niektórych parametrów modeli udało się znaleźć odpowiednie kombinacje dla lasów losowych oraz dla KNN, które wyglądały następująco:

- dla lasów losowych: `n_estimators=9`, `max_depth=15`, `criterion='absolute_error'`, `min_samples_split=4`;
- dla KNN: `n_neighbors=1`, `algorithm='kd_tree'`, `p=1`.

Trzeba zaznaczyć, że zmienione parametry modelu lasów losowych poprawiły zdolność modelu do wyjaśniania wariancji danych, co jest reprezentowane przez zwiększenie błędu R^2. Jednak



Rysunek 2: Wartości błędów po pierwszym eksperymencie.

błąd MAE się zwiększył, co może sugerować, że przewidywania modelu są mniej dokładne. Natomiast błąd MSE został bez zmian.

Dla obu sieci neuronowych wartości błędu R^2 dla różnych optymalizatorów dały rozmaite wyniki, co zostało pokazane w tabeli 2. Dla obu sieci najlepsze wyniki wyszły dla optymalizatora SGD. Jednocześnie dany optymalizator poprawił błąd R^2 w porównaniu do pierwszego eksperymentu.

Wybierając LSTM jako model, który najlepiej symuluje wartości współczynnika beta, zrobiłyśmy predykcje dla różnych 6 architektur i sprawdziłyśmy, który wypada najlepiej. Z wyników przedstawionych na rysunku 3 można stwierdzić, że model z jedną warstwą LSTM dla 256 neuronów, dwiema Dense z 256 neuronami i warstwą wyjściową Dense z 1 neuronami predykuje

Tabela 1: Wyniki dla KNN i lasów losowych przed i po zmianie:

kombinacja parametrów	R^2	MAE	MSE
domyślny KNN	-64.126504	0.112020	0.012684
zmieniony KNN	-59.724809	0.107823	0.011827
domyślne lasy losowe	-33.652777	0.081241	0.006749
zmienione lasy losowe	-23.886334	0.107824	0.011827

Tabela 2: Wynik błędu R^2 dla różnych optymalizatorów.

sieć neuronowa	Adadelta	Adagrad	Adam	RMSprop	SGD
MLP	-2280.17756	-449.965175	-20.007545	-25.75932	**-9.130922**
LSTM	-74.007351	-6.361244	-1.292562	-61.285785	**-0.86265**

najlepiej, a najgorsze wyniki wyszły dla architektury z najmniejszą ilością neuronów.

Wybierając Model 6. dla LSTM, wytrenowano go na różnej długości danych (historycznych). Przedstawione w tabeli 3 wyniki pokazują, że optymalnym czasem jest wybrany przez nas na samym początku okres 5 lat.

Tabela 3: Wartości błędów dla różnej wielkości danych historycznych.

okres	R^2	MAE	MSE
5 lat	-0.888023	0.052859	0.003049
10 lat	-417.427115	0.158164	0.025124
1 rok	-159.25669	0.202638	0.0412833
4 lat	-59.328707	0.103521	0.010815

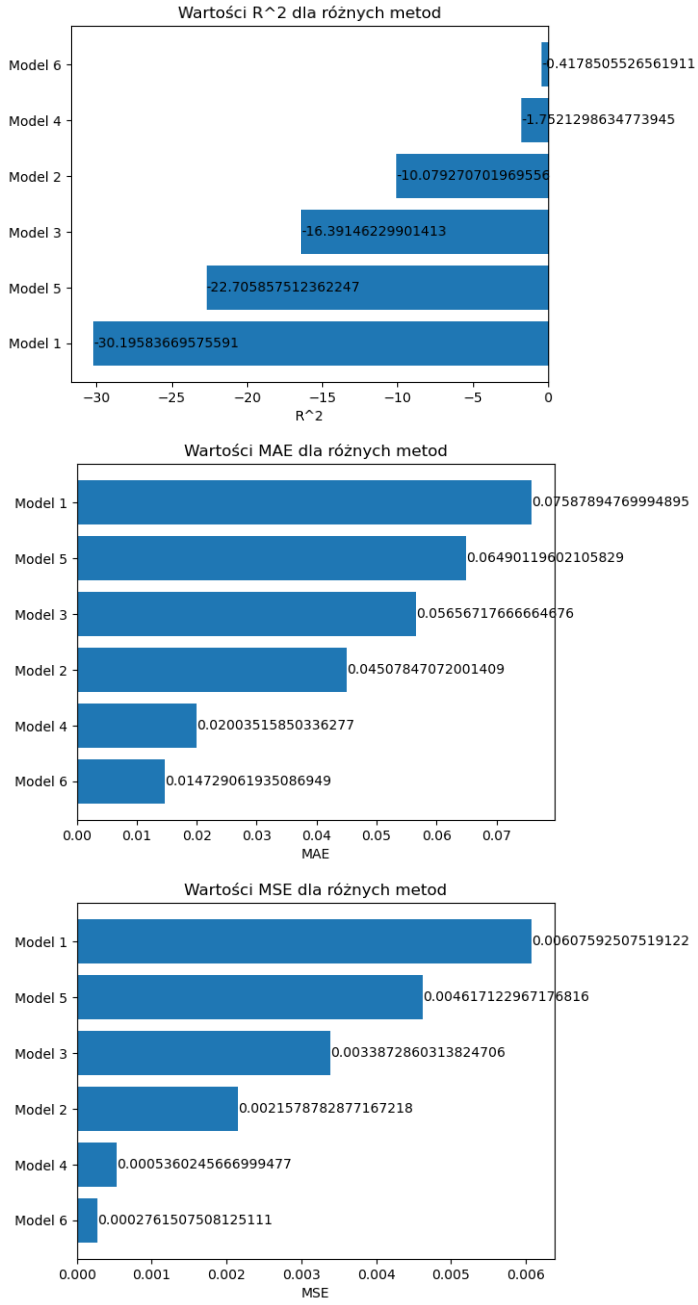
4 Podsumowanie i wnioski

4.1 Wnioski

Eksperymenty przeprowadzone w ramach tej pracy i wnioski wyciągnięte na ich podstawie umożliwiły na udzielenie odpowiedzi na pytania, które stanowiły główny cel badania. Wszystkie uzyskane wnioski zostały szczegółowo przedstawione i omówione w dalszej części tego rozdziału.

Wyniki przeprowadzonych eksperymentów zdają się zaprzeczać teorii, że niemożliwe jest przewidywanie współczynnika beta, gdyż opracowany model na podstawie danych z wcześniejszych lat umożliwił osiągnięcie nadzwyczajnie wysokich wyników predykowania współczynnika beta dla roku 2019. Ponadprzeciętna skuteczność generowanych predykcji nie wskazuje na dzieło przypadku w otrzymanych rezultatach.

Dodatkowo zostało potwierdzone, że jeśli chcemy symulować współczynnik beta, to lepiej nie opierać się na danych historycznych z więcej niż 5 lat.



Rysunek 3: Wartości błędów dla różnych architektur LSTM.

W oczekiwaniu na wyniki, rekurencyjne sieci neuronowe przewidywane były osiągać niższe lub porównywalne zyski w porównaniu do algorytmu losowego w kontekście predykcji współczynnika beta. Jednakże eksperymentalne rozwiązanie, które zostało opracowane, osiągnęło nadprzeciętny zysk przewyższający wyniki algorytmu losowego. Uzyskane wyniki jednoznacznie wskazują, że możliwe jest symulowanie współczynnika beta na podstawie historycznych danych spółek.

Po przeprowadzeniu analizy różnych potencjalnie atrakcyjnych architektur pod względem efektywności generowania predykcji wybrano najskuteczniejsze rozwiązanie. Okazało się nim struktura składająca się z jednej warstwy LSTM posiadająca 256 neuronów, dwóch warstw Dense dla 256 neuronów i warstwa wyjściowej Dense z jednym neuronem. Model ten osiągnął

błąd MAE równy 1,5% w porównaniu do wartości współczynnika beta z roku 2019 dla firmy Apple. Z tego powodu, ta metoda została uznana za najskuteczniejszą w kontekście symulacji współczynnika beta.

4.2 Podsumowanie

Podsumowując, w ramach badania przeprowadzono analizę predykcji wartości współczynnika beta dla firm Apple (AAPL) i rynku S&P 500 (SPY) na podstawie danych historycznych (lata 2014-2019).

Przeprowadzono eksperymenty, w których porównano różne modele uczenia maszynowego, takie jak regresja liniowa, lasy losowe, K najbliższych sąsiadów (KNN) oraz sieci neuronowe MLP i LSTM. Po analizie wyników najlepsze rezultaty uzyskano dla sieci neuronowych, zwłaszcza dla modelu LSTM.

Dla modeli lasów losowych i KNN przeprowadzono optymalizację parametrów, co pozwoliło poprawić niektóre miary jakości predykcji. Dla LSTM przetestowano różne architektury, a najlepsze wyniki uzyskano dla modelu zawierającego jedną warstwę LSTM z 256 neuronami, dwie warstwy Dense z 128 neuronami i ostatnią warstwę Dense z 1 neuronem.

Na koniec sprawdzono, jak różna długość danych historycznych wpływa na predykcje. Okazało się, że najlepsze wyniki uzyskuje się przy wykorzystaniu danych z ostatnich 5 lat.

W rezultacie przeprowadzonych eksperymentów udało się znaleźć efektywne modele predykcyjne dla wartości współczynnika beta, które mogą być wykorzystane w symulacjach finansowych dla firm i rynków.

Bibliografia

[1] W. Drobetz, F. Hollstein, T. Otto i M. Prokopczuk. „Estimating Security Betas via Machine Learning". W: *SSRN Electronic Journal* (2021).

[2] D. Hoang i K. Wiegratz. *Application of Machine Learning Algorithms for Simulating Beta Coefficient Change*. European Financial Management, John Wiley & Sons Ltd., 2022.

[3] F. Hollstein, M. Prokopczuk i C. Wese Simen. *Estimating beta: forecast adjustments and the impact of stock characteristics for a broad cross-section*. Journal of Financial Markets, 44. pp. 91-118, 2019.

[4] P. Ndikum. *Machine Learning Algorithms for Financial Asset Price Forecasting*. ArXiv, 2020.