# Klasyfikacja kwiatów Iris za pomocą uczenia maszynowego

Twoim zadaniem jest stworzenie modelu uczenia maszynowego, który będzie klasyfikował gatunki kwiatów Iris na podstawie ich cech morfologicznych. Zbiór danych Iris zawiera informacje o czterech cechach: długości i szerokości płatków oraz długości i szerokości działek kielicha dla trzech różnych gatunków: Iris setosa, Iris versicolor i Iris virginica.

<https://www.kaggle.com/datasets/uciml/iris>

**Zakres:**

* Przygotowanie danych: Zaimportuj zbiór danych Iris i podziel go na zbiór treningowy i testowy. Sprawdź, czy dane są zbalansowane pod względem liczby przykładów dla każdego gatunku.
* Eksploracja danych: Przeanalizuj dane, wykonując wizualizacje, takie jak wykresy punktowe i histogramy, aby zrozumieć relacje między cechami a klasami. Zidentyfikuj ewentualne zależności i różnice między gatunkami.
* Wybór modelu: Wybierz odpowiedni model klasyfikacji do zadania. Możesz rozważyć algorytmy takie jak K-nearest neighbors (k najbliższych sąsiadów), Support Vector Machines (SVM) lub Random Forests.
* Trenowanie modelu: Skonfiguruj model, wybierając odpowiednie hiperparametry i funkcję straty. Trenuj model na zbiorze treningowym i monitoruj postęp uczenia. Eksperymentuj z różnymi hiperparametrami, takimi jak liczba sąsiadów w KNN czy parametry kernela w SVM.
* Ocena modelu: Ocenić wydajność modelu na zbiorze testowym. Oblicz metryki oceny, takie jak dokładność klasyfikacji, precyzja, czułość i krzywa ROC. Porównaj wyniki dla różnych modeli i wybierz ten, który osiąga najlepsze wyniki.
* Interpretacja wyników: Przeanalizuj wyniki modelu i zastanów się, jakie cechy mają największy wpływ na klasyfikację gatunków Iris. Możesz wykorzystać miary ważności cech, takie jak współczynniki SVM lub ważności cech w Random Forests.
* Walidacja krzyżowa: Przeprowadź walidację krzyżową, aby sprawdzić stabilność i ogólną wydajność modelu. Oceń wydajność modelu na różnych podziałach danych treningowych i testowych.
* Optymalizacja modelu: możesz eksperymentować z różnymi technikami regularyzacji, skalowaniem danych czy technikami dostrajania hiperparametrów, takimi jak Grid Search lub Random Search.
* Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy.

# Rozpoznawanie cyfr za pomocą sieci neuronowej

Twoim zadaniem jest zaimplementowanie sieci neuronowej w języku Python, która będzie klasyfikować cyfry z zestawu danych MNIST. Zbiór danych MNIST składa się z 60 000 obrazów treningowych cyfr o rozmiarze 28x28 pikseli oraz 10 000 obrazów testowych.

<http://yann.lecun.com/exdb/mnist/>

**Zakres:**

* Przygotowanie danych: Zaimportuj zbiór danych MNIST i podziel go na zbiór treningowy i testowy. Znormalizuj wartości pikseli do zakresu od 0 do 1.
* Zdefiniowanie architektury sieci neuronowej: Stwórz model sieci neuronowej, wybierając odpowiednią liczbę warstw i liczby neuronów w każdej warstwie. Możesz użyć biblioteki Keras lub TensorFlow do tworzenia modelu. Możesz użyć DNN, CNN czy RMDL.
* Trenowanie modelu: Skonfiguruj model, wybierając funkcję straty, optymalizator i metryki oceny. Trenuj model na zbiorze treningowym za pomocą algorytmu wstecznej propagacji błędu. Eksperymentuj z różnymi hiperparametrami, takimi jak współczynnik uczenia i liczba epok.
* Ocena modelu: Ocenić wydajność modelu na zbiorze testowym. Oblicz metryki, takie jak dokładność klasyfikacji, precyzja, czułość i krzywą ROC.
* Optymalizacja modelu: możesz eksperymentować z różnymi architekturami sieci, hiperparametrami, regularyzacją i technikami dostrajania modelu, takimi jak Dropout czy Batch Normalization.
* Dokumentacja: Zadbaj o utworzenie czytelnej dokumentacji kodu, która opisuje architekturę sieci, proces trenowania i oceny modelu.
* Podsumowanie: Przygotuj podsumowanie swojej pracy, w którym omówisz osiągnięte wyniki, wyzwania, z którymi się spotkałeś, oraz możliwe sposoby poprawy wydajności modelu.

# System ekspertowy oparty na wnioskowaniu rozmytym

Twoim zadaniem jest stworzenie prostego systemu ekspertowego opartego na wnioskowaniu rozmytym w języku Python. System ma na celu ocenę przydatności różnych potraw w zależności od ich cech rozmytych, takich jak smak, pikantność, konsystencja i słodycz.

**Zakres:**

* Zdefiniuj zbiór reguł: Opracuj zbiór reguł, które opisują ocenę przydatności potraw na podstawie różnych cech rozmytych. Na przykład, "Jeśli smak jest bardzo dobry i pikantność jest umiarkowana, to potrawa jest bardzo przydatna".
* Zdefiniuj zbiór danych: Stwórz zestaw danych obejmujący cechy rozmyte potraw, takie jak smak, pikantność, konsystencja i słodycz. Możesz zdefiniować te cechy za pomocą funkcji przynależności rozmytej, takich jak trójkątne, trapezoidalne lub gaussowskie.
* Implementuj wnioskowanie rozmyte: Zaimplementuj mechanizm wnioskowania rozmytego, który używa zdefiniowanych reguł i danych wejściowych, aby ocenić przydatność potraw. Wykorzystaj algorytmy wnioskowania rozmytego, takie jak Minimum lub Maximum, aby uzyskać ocenę.
* Przygotuj interfejs użytkownika: Stwórz prosty interfejs użytkownika w konsoli lub za pomocą biblioteki graficznej, który umożliwi użytkownikowi podanie cech rozmytych potrawy. Następnie system powinien obliczyć ocenę przydatności potrawy na podstawie wprowadzonych danych.
* Testowanie i ewaluacja: Przetestuj system, wprowadzając różne kombinacje cech rozmytych potraw i sprawdź, czy ocena przydatności jest zgodna z oczekiwaniami na podstawie zdefiniowanych reguł.
* Optymalizacja: możesz eksperymentować z różnymi regułami, funkcjami przynależności czy algorytmami wnioskowania w celu poprawy wydajności i trafności systemu.
* Dokumentacja: Przygotuj dokumentację, która opisuje architekturę systemu, reguły, użyte funkcje przynależności, wyniki testów i ewentualne wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy.

# Optymalizacja trasy dostaw za pomocą algorytmu genetycznego

Twoim zadaniem jest zastosowanie algorytmu genetycznego w celu optymalizacji trasy dostaw dla przedsiębiorstwa kurierskiego. Przedsiębiorstwo posiada zestaw klientów, którzy mają określone lokalizacje oraz pakunki do dostarczenia. Celem jest znalezienie optymalnej trasy, która minimalizuje łączny koszt lub czas dostawy.

**Założenia:**

1. Przyjmij, że firma kurierska ma 10 klientów, oznaczonych jako K1, K2, ..., K10. Każdy klient ma przypisaną lokalizację, którą można przedstawić jako współrzędne geograficzne (np. szerokość i długość geograficzną).
2. Przypisz koszt lub czas dostawy dla każdej pary klientów. Na przykład, jeśli masz dane kosztu podróży między klientami w postaci macierzy kosztów, gdzie wartość w komórce (i, j) reprezentuje koszt podróży od klienta i do klienta j, możesz utworzyć taką macierz kosztów dla wszystkich par klientów.

**Zakres:**

* Zdefiniuj kodowanie: Stwórz odpowiednie kodowanie genotypu, które reprezentuje trasę dostaw jako sekwencję odwiedzanych klientów. Możesz wybrać kodowanie permutacyjne, w którym każdy gen odpowiada innemu klientowi.
* Inicjalizacja populacji: Wygeneruj początkową populację chromosomów, reprezentujących różne trasy dostaw. Populacja powinna zawierać losowo wygenerowane trasy. Przykładowo wygeneruj początkową populację chromosomów, które reprezentują różne trasy dostaw. Na przykład, możesz wygenerować 100 chromosomów, z których każdy jest permutacją K1, K2, ..., K10.
* Ocena funkcji celu: Zaimplementuj funkcję celu, która ocenia jakość trasy dostaw. Może to być łączny koszt dostawy, czas podróży, zużycie paliwa itp. Ocena powinna być przypisana każdemu chromosomowi w populacji. Przykładowo funkcja celu może sumować koszty podróży między klientami na trasie.
* Operatory genetyczne: Zdefiniuj operatory genetyczne, takie jak **selekcja, krzyżowanie i mutacja**, które będą stosowane na populacji w celu tworzenia nowych pokoleń. Możesz użyć popularnych technik, takich jak selekcja turniejowa, krzyżowanie jednopunktowe lub wielopunktowe oraz mutacja zmiany miasta.
* Proces ewolucji: Wykonuj proces ewolucji, w którym kolejne pokolenia populacji są generowane poprzez zastosowanie operacji genetycznych. Następnie oceniaj jakość każdego nowego pokolenia i ewentualnie stosuj strategie elitaryzmu, aby zachować najlepsze rozwiązania.
* Warunki zatrzymania: Zdefiniuj warunki zatrzymania algorytmu genetycznego. Może to być maksymalna liczba generacji, osiągnięcie satysfakcjonującego rozwiązania lub brak znaczącej poprawy przez określoną liczbę pokoleń.
* Prezentacja wyników: Po zakończeniu algorytmu genetycznego, przedstaw optymalną trasę dostaw, która minimalizuje koszt lub czas podróży. Możesz również wizualizować trasę na mapie dla lepszej prezentacji.
* Optymalizacja i eksperymenty: możesz przeprowadzić eksperymenty z różnymi parametrami algorytmu genetycznego, takimi jak rozmiar populacji, współczynnik krzyżowania i mutacji, aby znaleźć optymalne ustawienia dla danego problemu.
* Dokumentacja: Przygotuj dokumentację, która opisuje proces optymalizacji, użyte operatory genetyczne, wyniki eksperymentów i ocenę jakości optymalnej trasy dostaw.

# Klasyfikacja Danych z Użyciem Regresji Logistycznej

* **Opis**: Stwórz model klasyfikacji binarnej, który na podstawie dostępnych danych będzie przewidywał, do której klasy należy dany rekord. Wykorzystaj regresję logistyczną.
* **Zakres**:
  + Przygotowanie danych (np. dane o klientach banku, danych medycznych itp.)
  + Podział na zbiór treningowy i testowy
  + Stworzenie modelu regresji logistycznej
  + Walidacja modelu (metryki: accuracy, precision, recall, F1-score)
  + Wykorzystanie cross-walidacji lub regularyzacji.
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy.

**Dane**: Możesz skorzystać z popularnych zbiorów danych dostępnych na platformie **UCI Machine Learning Repository**:

* Titanic: Machine Learning from Disaster (Kaggle)
* [Breast Cancer Wisconsin (UCI)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic))

# Algorytm Klasyfikacji Tekstu z Użyciem NLP

* **Opis**: Zaimplementuj algorytm klasyfikacji tekstu (np. klasyfikacja wiadomości e-mail na spam/niespam) z użyciem technik NLP.
* **Zakres**:
  + Zbiór danych (np. zbiór wiadomości e-mail, recenzje produktów)
  + Przetwarzanie tekstu (tokenizacja, lematyzacja, usuwanie stop-słów)
  + Klasyfikacja za pomocą algorytmu Naive Bayes lub SVM
  + Ewaluacja wyników
  + Eksperyment z innymi algorytmami klasyfikacji oraz technikami przetwarzania tekstu.
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy

**Dane**:

* Spam Email Dataset (Kaggle)
* 20 Newsgroups Dataset (scikit-learn) - klasyfikacja artykułów z różnych grup dyskusyjnych
* IMDb Reviews Dataset (Kaggle)

# Rozpoznawanie Obrazów z Użyciem Sztucznych Sieci Neuronowych (CNN)

* **Opis**: Zaimplementuj prosty model konwolucyjnej sieci neuronowej (CNN), który rozpoznaje obrazy z jednego z dostępnych zbiorów danych (np. CIFAR-10, MNIST).
* **Zakres**:
  + Przygotowanie danych (normalizacja, augmentacja)
  + Stworzenie architektury sieci CNN (warstwy konwolucyjne, pooling, fully connected)
  + Trenowanie modelu i walidacja wyników
  + Dodanie zaawansowanych technik, jak dropout, batch normalization.
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy

**Dane**:

* CIFAR-10 Dataset - 10 klas obrazów
* MNIST Dataset - zbiór ręcznie pisanych cyfr
* Fashion MNIST (Kaggle)

# Algorytm Genetyczny do Rozwiązywania Problemów Optymalizacyjnych

* **Opis**: Zaimplementuj algorytm genetyczny, który rozwiązuje prosty problem optymalizacyjny (np. problem plecakowy, minimalizacja funkcji matematycznej).
* **Zakres**:
  + Kodowanie problemu
  + Implementacja operatorów genetycznych (selekcja, krzyżowanie, mutacja)
  + Ewaluacja rozwiązań (fitness function)
  + Dodanie zaawansowanych operatorów genetycznych lub optymalizacji parametrów algorytmu.
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy
* **Dane**: Algorytm genetyczny zwykle działa na sztucznie generowanych danych

# Klasteryzacja Danych za Pomocą Algorytmu K-Means

* **Opis**: Wykorzystaj algorytm K-Means do podziału danych na grupy (klastry) na podstawie pewnych cech.
* **Zakres**:
  + Wybór i przygotowanie zbioru danych (np. dane demograficzne, dane o zakupach klientów)
  + Implementacja algorytmu K-Means
  + Wizualizacja wyników (np. za pomocą PCA)
  + Testowanie algorytmu DBSCAN jako alternatywnej metody klasteryzacji.
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy

**Dane**:

* Mall Customers Dataset (Kaggle)
* [Wholesale Customers Dataset (UCI)](https://archive.ics.uci.edu/ml/datasets/wholesale+customers)

# Budowa Systemu Rekomendacji z Użyciem Algorytmów Filtracji Kolaboracyjnej

* **Opis**: Stwórz prosty system rekomendacji, który będzie rekomendował użytkownikom produkty na podstawie historii ocen innych użytkowników.
* **Zakres**:
  + Przygotowanie zbioru danych (np. MovieLens)
  + Implementacja filtracji kolaboracyjnej (na podstawie podobieństwa użytkowników lub produktów)
  + Ocena systemu rekomendacji (precision, recall)
  + Wprowadzenie macierzy faktoryzacji (matrix factorization).
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy

**Dane**:

* MovieLens Dataset - zbiór danych ocen filmów użytkowników
* Amazon Product Reviews (Kaggle)

# Predykcja Czasu Życia Klienta (Customer Lifetime Value)

* **Opis**: Wykorzystaj algorytmy uczenia maszynowego, aby przewidzieć wartość klienta (Customer Lifetime Value) w oparciu o dane dotyczące zakupów klientów.
* **Zakres**:
  + Przygotowanie i eksploracja danych (np. transakcje zakupowe, dane demograficzne)
  + Wybór odpowiedniego modelu (regresja, drzewa decyzyjne)
  + Ewaluacja modelu
  + Wykorzystanie bardziej zaawansowanych modeli, jak XGBoost czy Random Forest.
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy

**Dane**:

* [Online Retail Dataset (UCI)](https://archive.ics.uci.edu/ml/datasets/Online+Retail)
* Customer Segmentation (Kaggle)

# Detekcja Anomalii w Danych z Użyciem Autoenkoderów

* **Opis**: Zaimplementuj autoenkoder do detekcji anomalii w danych (np. wykrywanie oszustw w transakcjach finansowych).
* **Zakres**:
  + Przygotowanie zbioru danych (transakcje finansowe, dane sensoryczne)
  + Budowa autoenkodera i trenowanie modelu
  + Detekcja anomalii na podstawie błędu rekonstrukcji
  + Porównanie z tradycyjnymi metodami detekcji anomalii (np. Isolation Forest).
  + Dokumentacja: Przygotuj dokumentację opisującą etapy procesu, parametry modelu, otrzymane wyniki i wnioski. Zadbaj o czytelną i zrozumiałą prezentację swojej pracy

**Dane**:

* Credit Card Fraud Detection (Kaggle) - wykrywanie oszustw w transakcjach kart kredytowych
* [KDD Cup 1999 Data (UCI)](http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html) - zbiór danych dotyczący wykrywania ataków sieciowych

# Główne źródła danych:

* [**UCI Machine Learning Repository**](https://archive.ics.uci.edu/ml/index.php) – zawiera szeroką gamę różnych zestawów danych do uczenia maszynowego.
* **Kaggle Datasets** – Kaggle to platforma do współpracy nad projektami AI, oferuje mnóstwo otwartych zbiorów danych z różnych dziedzin.
* **scikit-learn Datasets** – scikit-learn ma wbudowane popularne zbiory danych, które można łatwo załadować w Pythonie.