Tema 2 IA

Descriere Generală

1. **Predicția popularității știrilor** (clasificare multi-class cu 5 clase)
2. **Predicția riscului de boală coronariană** (clasificare binară)

Structura Proiectului

```
ml_pipeline/
   data_explorer.py          # Analiza exploratorie a datelor (EDA)
   data_processor.py         # Preprocesarea și curățarea datelor
   logistic_regression.py    # Implementare custom regresie logistică
   ml_orchestrator.py        # Antrenarea și evaluarea modelelor
   run_pipeline.py          # Pipeline-ul complet
   README.md               # Acest fișier
```

1. Explorarea Datelor (EDA)

1.1 Analiza Tipului de Atribute și Plajei de Valori

Implementare în `data_explorer.py`:

```python
def categorize_columns(df, target_col=None, threshold=20):
    """Separate columns into numeric and categorical types"""
    # Separarea coloanelor în numerice și categorice
    # Distincția între continue și discrete
```

Abordare: - **Atribute numerice continue**: Identificate prin `select_dtypes(include=['number'])` și filtrate după numărul de valori unice (>= 20) - **Atribute discrete/ordinale**: Atribute numerice cu puține valori unice (< 20) - **Atribute categorice**: Toate coloanele non-numerice

1.1.1 Statistici pentru Atribute Numerice Continue News Dataset - Exemple de atribute continue:

Atribut	Non-Null	Medie	Std Dev	Min	Q1 (25%)	Median (50%)	Q3 (75%)	Max
days_since_published	31715	354.2	214.3	8.0	164.0	339.0	542.0	731.0
title_word_count	31715	10.4	2.0	2.0	9.0	10.0	12.0	23.0
content_word_count	31715	546.4	471.8	0.0	246.0	409.0	716.0	8474.0
unique_word_ratio	31715	0.548	0.127	0.0	0.471	0.539	0.608	1.0
external_links	31715	10.9	69.2	0.0	1.0	3.0	9.0	4589.0
avg_word_length	31715	4.55	0.44	0.0	4.36	4.54	4.74	8.04
engagement_ratio	31715	2847.1	11584.2	0.0	58.0	174.9	1076.3	284216.7

Heart Dataset - Exemple de atribute continue:

Atribut	Non-Null	Medie	Std Dev	Min	Q1 (25%)	Median (50%)	Q3 (75%)	Max
age	3392	49.6	8.6	32.0	42.0	49.0	56.0	70.0
systolic_pressure	3392	132.4	22.0	83.5	117.0	128.5	144.0	295.0
diastolic_pressure	3392	83.0	12.0	48.0	75.0	81.0	90.0	142.5
daily_cigarettes	3392	9.0	11.9	0.0	0.0	0.0	20.0	70.0
cholesterol_level	3392	236.7	44.2	113.0	206.0	234.0	263.0	696.0
heart_rate	3392	75.9	12.0	44.0	68.0	75.0	83.0	143.0
mass_index	3392	25.8	4.1	15.5	23.1	25.3	28.1	56.8
blood_sugar_level	3392	87.1	23.4	65.0	71.0	81.0	95.0	394.0

1.1.2 Statistici pentru Atribute Discrete/Ordinale News Dataset - Atribute discrete:

Atribut	Non-Null	Valori Unice	Observații
internal_links	31715	9	0-8 legături interne
keyword_worst_min_shares	31715	3	Categorii de popularitate
min_positive_sentiment	31715	11	Scale sentiment 0.0-1.0
channel_lifestyle	28544	2	Da/Nu (9.99% valori lipsă)
channel_entertainment	31715	2	Da/Nu
publication_period	31715	2	Weekday/Weekend

Heart Dataset - Atribute discrete:

Atribut	Non-Null	Valori Unice	Observații
gender	3392	2	M/F
education_level	3392	4	1-4 scale
blood_pressure_medication	3392	2	Da/Nu
smoking_status	3392	2	Da/Nu
hypertension_history	3392	2	Da/Nu
stroke_history	3392	2	Da/Nu
diabetes_history	3392	2	Da/Nu
high_blood_sugar	3392	2	Da/Nu

1.1.2 Statistici pentru Atribute Discrete/Ordinale News Dataset - Atribute discrete:

Atribut	Non-Null	Valori Unice	Observații
internal_links	31715	9	0-8 legături interne
keyword_worst_min_shares	31715	3	Categorii de popularitate
min_positive_sentiment	31715	11	Scale sentiment 0.0-1.0

Atribut	Non-Null	Valori Unice	Observații
channel_lifestyle	28544	2	Da/Nu (9.99% valori lipsă)
channel_entertainment	31715	2	Da/Nu
publication_period	31715	2	Weekday/Weekend

Heart Dataset - Atribute discrete:

Atribut	Non-Null	Valori Unice	Observații
gender	3392	2	M/F
education_level	3392	4	1-4 scale
blood_pressure_medication	3392	2	Da/Nu
smoking_status	3392	2	Da/Nu
hypertension_history	3392	2	Da/Nu
stroke_history	3392	2	Da/Nu
diabetes_history	3392	2	Da/Nu
high_blood_sugar	3392	2	Da/Nu

Observații importante: - **News Dataset**: Atributul `external_links` prezintă outliers extremi (max: 4589 vs Q3: 9) - **News Dataset**: `channel_lifestyle` are 9.99% valori lipsă necesitând imputare - **Heart Dataset**: Toate atributele sunt complete (fără valori lipsă) - **Heart Dataset**: Predomină atributele binare medicale (Da/Nu pentru condiții)

1.1.3 Vizualizări Boxplot și Histograme Boxplots pentru Atribute Numerice Continue:

News Dataset Boxplots *News Dataset: Distribuția atributelor numerice continue cu identificarea outliers*

Heart Dataset Boxplots *Heart Dataset: Distribuția atributelor numerice continue - observați outliers în cholesterol_level și total_cigarettes*

Histograme pentru Atribute Categorice:

News Dataset Histograms *News Dataset: Distribuția atributelor categorice - majoritatea sunt binare (Y/N) cu distribuții echilibrate*

Heart Dataset Histograms *Heart Dataset: Distribuția atributelor categorice - predomină valorile binare pentru condițiile medicale*

Vizualizări implementate: - **Boxplots** pentru atribute numerice: `create_boxplot_grid()` - permite identificarea valorilor extreme - **Histograme** pentru atribute categorice: `create_histogram_grid()` - arată distribuția valorilor

1.2 Analiza Echilibrului de Clase

```
def plot_target_distribution(train_df, test_df, target_col, dataset_name):
    """Compare target distribution between train and test sets"""
```

News Dataset Class Distribution *News Dataset: Distribuția claselor în train vs test - evident dezechilibrul cu "Slightly Popular" dominant (47.9%)*

Rezultate observate: - **News Dataset**: Severe dezechilibrat - "Slightly Popular" (47.9%) vs "Unpopular" (2.7%) - Slightly Popular: 15,194 / 3,799 (train/test)
- Moderately Popular: 9,605 / 2,401 - Popular: 4,297 / 1,074 - Viral: 1,748 / 437 - Unpopular: 871 / 218 (cel mai mic)

- **Heart Dataset**: Moderat dezechilibrat - Clasa 0 (84.8%) vs Clasa 1 (15.2%)
  - Fără risc (0): 2,877 / 719 (train/test)
  - Cu risc (1): 515 / 129

Impact asupra modelelor: Pentru seturile dezechilibrate am implementat ponderarea claselor în algoritmi pentru a compensa bias-ul către clasele majoritare.

1.3 Analiza Corelației între Atribute

Pentru atribute numerice:

```
def compute_correlation_matrix(df, columns):
    """Calculate correlation matrix for numeric columns"""
    return df[columns].corr(method='pearson')
```

Pentru atribute categorice:

```
def compute_chi_square_matrix(df, columns):
    """Calculate chi-square test p-values for categorical columns"""
```

1.3.1 Corelații Numerice - Heart Dataset Heart Dataset Numeric Correlations *Heart Dataset: Matrice de corelație Pearson - observați corelațiile puternice între daily_cigarettes și total_cigarettes (0.98)*

Corelații semnificative identificate: - **daily_cigarettes total_cigarettes**: r = 0.98 (corelație aproape perfectă) - **systolic_pressure diastolic_pressure**: r = 0.79 (corelație puternică așteptată) - **blood_sugar_level glucose**: r = 1.0 (identice - redundanță completă) - **mass_index diastolic_pressure**: r = 0.37 (corelație moderată)

1.3.2 Independența Categorică - Chi-Square Analysis News Dataset Categorical Independence *News Dataset: Analiza independenței categorice - valorile p mai mici (roșu închis) indică dependență statistică*

Heart Dataset Categorical Independence *Heart Dataset: Analiza independenței categorice - majoritatea variabilelor medicale sunt independente*

Observații din testul Chi-pătrat: - **News Dataset**: Dependențe puternice între zilele săptămânii ($p < 0.05$) - **Heart Dataset**: Variabilele medicale sunt în general independente - **Variabilele url** din News Dataset eliminate automat din cauza cardinalității mari

Abordare: - Corelația Pearson pentru atribute numerice (identifică redundanța liniară) - Testul Chi-square pentru atribute categorice (testează independența statistică) - **Eliminare automată** a features cu corelație > 0.85 sau dependență $p < 0.05$

2. Preprocesarea Datelor

2.1 Date Lipsă

Implementare în `data_processor.py`:

```python
def fill_missing_values(df, numeric_strategy='median', text_strategy='most_frequent'):
    """Fill missing values using specified strategies"""
```

Strategii alese: - **Atribute numerice**: Mediană (robustă la valori extreme) - **Atribute categorice**: Valoarea cea mai frecventă

2.2 Valori Extreme (Outliers)

```python
def find_outliers_iqr(df, columns, multiplier=1.5):
    """Detect outliers using IQR method"""
    # Q1 - 1.5*IQR și Q3 + 1.5*IQR
```

Metodă: Intervalul interquartil (IQR) cu factorul 1.5 **Tratament**: Înlocuirea cu mediana pentru stabilitate

2.3 Eliminarea Atributelor Redundante

```python
def find_correlated_features(correlation_matrix, threshold=0.85):
    """Find highly correlated features to remove"""


def find_dependent_categorical_features(p_value_matrix, significance_level=0.05):
    """Find categorical features that are dependent"""
```

Criterii de eliminare: - Corelație Pearson > 0.85 pentru atribute numerice - p-value < 0.05 în testul Chi-pătrat pentru atribute categorice

2.4 Standardizarea Datelor

```python
def standardize_features(df, columns):
    """Standardize numeric features"""
    scaler = StandardScaler()
```

Necesitate: Atributele numerice au scale diferite (ex: zile vs. număr de cuvinte)
Metodă: StandardScaler (medie=0, deviație=1)

3. Utilizarea Algoritmilor de Învățare Automată

3.1 Arbori de Decizie

Hiperparametri configurați:

```
news_tree_config = {
    'max_depth': 8,           # Previne overfitting
    'min_samples_leaf': 10,   # Stabilitate predicții
    'min_samples_split': 20,  # Control granularitate
    'max_features': 'sqrt',   # Reducere dimensionalitate
    'criterion': 'gini',      # Măsură impuritate
    'random_state': 42        # Reproductibilitate
}

# Heart Dataset
heart_tree_config = {
    'max_depth': 6,           # Adâncime redusă (dataset mic)
    'min_samples_leaf': 5,
    'min_samples_split': 10,
    'max_features': 'sqrt',
    'criterion': 'gini',
    'random_state': 42
}
```

Justificare alegeri: - `max_depth` mai mică pentru Heart dataset (dataset mai mic, risc overfitting) - `min_samples_leaf` mai mare pentru stabilitate - `gini` pentru performanță computațională

3.2 Păduri Aleatoare - 1.5 puncte

Hiperparametri configurați:

```
news_forest_config = {
    'n_estimators': 20,       # Balans performanță/timp
    'max_depth': 10,          # Puțin mai adânc decât Decision Tree
    'min_samples_leaf': 5,
    'min_samples_split': 10,
    'max_features': 'sqrt',   # Diversitate arbori
    'bootstrap': True,        # Sampling cu revenire
    'n_jobs': -1             # Paralelizare
}
```

Avantaje observate: - Performanță superioară față de Decision Tree individual - Robustețe la overfitting prin ensemble

3.3 Regresie Logistică

Implementare manuală în `logistic_regression.py`:

Implementare Binară:

```
class CustomLogisticRegression(BaseEstimator, ClassifierMixin):
    def __init__(self, learning_rate=0.01, max_epochs=1000, class_weights=None):
        # Funcția sigmoid cu stabilitate numerică
        # Gradient descent cu ponderare clase
```

Implementare Multi-class:

```
class MultiClassLogisticRegression(BaseEstimator, ClassifierMixin):
    # Strategia One-vs-Rest pentru clasificare multi-class
```

Configurație:

```
# News Dataset (Multi-class)
news_logistic_config = {
    'learning_rate': 0.01,
    'max_epochs': 1000,
    'class_weights': 'balanced'  # Compensează dezechilibrul
}

# Heart Dataset (Binary)
heart_logistic_config = {
    'learning_rate': 0.01,
    'max_epochs': 1000,
    'class_weights': {0: 1.0, 1: 3.0}  # Accent pe clasa pozitivă
}
```

Aspecte tehnice: - Implementare sigmoid cu clipping pentru stabilitate numerică - Gradient descent cu learning rate adaptat - Ponderarea claselor pentru dataset dezechilibrat - Strategia One-vs-Rest pentru multi-class

3.4 Multi-Layered Perceptron (MLP)

Configurație:

```
# News Dataset
news_mlp_config = {
    'hidden_layer_sizes': (100, 50),  # 2 straturi ascunse
    'activation': 'relu',             # Funcție activare
    'solver': 'adam',                 # Optimizator adapativ
    'alpha': 0.001,                   # Regularizare L2
    'learning_rate_init': 0.001,      # Learning rate inițial
    'max_iter': 500,                  # Număr epoci
    'early_stopping': True,           # Prevenire overfitting
```

```
    'random_state': 42
}

# Heart Dataset
heart_mlp_config = {
    'hidden_layer_sizes': (50, 25),    # Arhitectură mai mică
    'activation': 'relu',
    'solver': 'adam',
    'alpha': 0.001,
    'learning_rate_init': 0.001,
    'max_iter': 500,
    'early_stopping': True,
    'random_state': 42
}
```

Justificare arhitectură: - Heart dataset: Arhitectură mai mică (50, 25) pentru dataset redus - News dataset: Arhitectură mai complexă (100, 50) pentru complexitatea problemei - Early stopping pentru prevenirea overfitting-ului

3.4.1 Analiza Curbelor de Antrenare MLP News Dataset - Curbe de Learning: - **Training Accuracy**: Creștere graduală de la ~45% la ~78% în primele 200 epoci - **Validation Accuracy**: Pattern similar cu training, indicator de generalizare bună - **Training Loss**: Scădere exponențială de la ~1.6 la ~0.6 - **Validation Loss**: Scădere paralelă cu training loss, early stopping la epoca 312 - **Concluzie**: Model bine calibrat, fără overfitting evident

Heart Dataset - Curbe de Learning: - **Training Accuracy**: Creștere rapidă la ~85% în primele 50 epoci - **Validation Accuracy**: Stabilizare la ~84%, convergență bună - **Training Loss**: Scădere de la ~0.7 la ~0.4 - **Validation Loss**: Pattern stabil, early stopping la epoca 156 - **Concluzie**: Convergență rapidă, dataset mai simplu

Observații importante: - **Nu există semne de overfitting** în niciun caz - **Early stopping** funcționează eficient - **Validation curves** urmăresc training curves îndeaproape

4. Pipeline de Preprocessing

Implementare în `ml_orchestrator.py`:

```
def create_preprocessing_pipeline(X_data):
    # OneHotEncoder pentru variabile categorice
    # StandardScaler pentru variabile numerice
    # Gestionarea cardinalității mari
```

Aspecte importante: - **OneHotEncoder** cu limitări: `max_categories=20`, `min_frequency=0.01` - **Excluderea** atributelor cu cardinalitate mare (ex: URL-uri) - **ColumnTransformer** pentru aplicarea diferită pe tipuri de date

5. Evaluarea Algoritmilor

Rezultate News Dataset:

Model	Accuracy	Training Time
Neural Network	**0.7732**	3.62s
Random Forest	0.6311	0.75s
Custom Logistic Regression	0.5281	13.48s
Decision Tree	0.5226	0.36s

News Dataset Performance Comparison *News Dataset: Comparația performanțelor - Neural Network dominăy*

Rezultate Heart Dataset:

Model	Accuracy	Training Time
Random Forest	**0.8491**	0.10s
Neural Network	**0.8443**	0.27s
Decision Tree	0.8420	0.01s
Custom Logistic Regression	0.7889	0.21s

Heart Dataset Performance Comparison *Heart Dataset: Performanțe apropiate pentru primele 3 modele, dar Custom Logistic are cea mai bună utilitate clinică*

5. Analiza Detaliată a Metricilor per Clasă

5.1 News Dataset - Metrici Detaliate

Model	Metric	Moderately Popular	Popular	Slightly Popular	Unpopular	Viral	**Overall Accuracy**
Neural Network	**Precision**	**0.81**	0.59	**0.80**	**0.70**	**0.45**	**0.7732**
	Recall	**0.82**	**0.54**	**0.91**	**0.23**	**0.15**	
	F1-Score	**0.82**	**0.57**	**0.86**	**0.35**	**0.23**	
Random Forest	**Precision**	0.66	**0.65**	0.62	0.00	**1.00**	0.6311
	Recall	0.57	0.03	**0.95**	0.00	0.01	
	F1-Score	0.61	0.06	**0.75**	0.00	0.02	

Model	Metric	Moderately Popular	Popular	Slightly Popular	Unpopular	Viral	**Overall Accuracy**
Decision Tree	**Precision**	0.47	0.29	0.54	0.14	0.00	0.5226
	Recall	0.36	0.03	**0.85**	0.00	0.00	
	F1-Score	0.41	0.05	0.66	0.01	0.00	
Custom Logistic Regression	**Precision**	0.58	0.28	**0.68**	0.17	0.13	0.5281
	Recall	0.49	0.18	0.69	**0.32**	**0.30**	
	F1-Score	0.53	0.22	**0.69**	**0.22**	**0.18**	

5.2 Heart Dataset - Metrici Detaliate

Model	Metric	No Risk (0)	High Risk (1)	**Overall Accuracy**
Random Forest	**Precision**	**0.85**	**1.00**	**0.8491**
	Recall	**1.00**	0.01	
	F1-Score	**0.92**	0.02	
Neural Network	**Precision**	**0.85**	0.40	**0.8443**
	Recall	**0.99**	0.05	
	F1-Score	**0.91**	0.08	
Decision Tree	**Precision**	**0.85**	0.14	0.8420
	Recall	**0.99**	0.01	
	F1-Score	**0.91**	0.01	
Custom Logistic Regression	**Precision**	**0.88**	0.31	0.7889
	Recall	0.87	**0.32**	

Model	Metric	No Risk (0)	High Risk (1)	**Overall Accuracy**
	F1-Score	**0.88**	**0.31**	

6. Analiza Detaliată a Confusion Matrices

6.1 News Dataset (Multi-class Classification)

Neural Network - Cea Mai Bună Performanță (77.32%) Neural Network Confusion Matrix - News *Neural Network: Cel mai echilibrat model cu detecția decentă a tuturor claselor*

Puncte forte: - **Moderately Popular**: 1961/2401 = 81.7% recall - excelent - **Popular**: 579/1074 = 53.9% recall - decent pentru clasa dificilă - **Slightly Popular**: 3474/3799 = 91.4% recall - foarte bun - **Unpopular**: 51/218 = 23.4% recall - cel mai bun dintre toate modelele - **Viral**: 66/437 = 15.1% recall - dificil, dar mai bun decât restul

Random Forest - Performanță Solidă (63.11%) Random Forest Confusion Matrix - News *Random Forest: Bias extrem către "Slightly Popular", eșec la clasele minoritare*

Puncte forte: - **Slightly Popular**: 3607/3799 = 94.9% recall - excelent - **Moderately Popular**: 1362/2401 = 56.7% recall - decent

Puncte slabe: - **Popular**: 31/1074 = 2.9% recall - aproape zero detectare - **Unpopular**: 0/218 = 0% recall - eșec - **Viral**: 4/437 = 0.9% recall - aproape zero detectare

Custom Logistic Regression - Distribuție Mai Echilibrată (52.81%) Custom Logistic Regression Confusion Matrix - News *Custom Logistic Regression: Singura care încearcă să detecteze toate clasele echilibrat*

Puncte forte: - **Unpopular**: 71/218 = 32.6% recall - surprinzător de bun - **Viral**: 126/437 = 28.8% recall - cel mai bun recall pentru această clasă - Singura care încearcă să detecteze toate clasele

Puncte slabe: - Acuratețe generală mai mică din cauza confuziei între clase - Prea multe false positive pentru clasele minoritare

6.2 Heart Dataset (Binary Classification)

Random Forest - Cel Mai Mare Accuracy (84.91%) Random Forest Confusion Matrix - Heart *Random Forest: Accuracy înalt dar bias extrem către clasa majoritară*

Problemă majoră: Clasifică aproape totul ca "fără risc" - **Impact clinic**: Pacienții cu risc sunt nedetectați!

Custom Logistic Regression - Cel Mai Echilibrat (78.89%) Custom Logistic Regression Confusion Matrix - Heart *Custom Logistic Regression: Trade-off acceptabil între accuracy și detectarea riscului*

Puncte forte: - **Cel mai bun recall pentru clasa pozitivă**: 31.8% - Balans mai bun între sensitivitate și specificitate

Neural Network - Performanță Intermediară (84.43%) Neural Network Confusion Matrix - Heart *Neural Network: Accuracy ridicat dar bias similar cu Random Forest*

Observații similare cu Random Forest - preferință pentru clasa majoritară

6.3 Impactul Dezechilibrului de Clase

News Dataset:

- **Slightly Popular**: 47.9% din date → toate modelele bias către aceasta
- **Unpopular**: 2.7% din date → ignorată de majoritatea modelelor
- **Soluția**: Neural Network reușește prin capacitatea de învățare non-liniară

Heart Dataset:

- **Fără risc (0)**: 84.8% din date
- **Cu risc (1)**: 15.2% din date
- **Problemă critică**: Majoritatea modelelor sacrifică recall-ul pentru acuratețe

6.4 Concluzii

Pentru News Dataset:

1. **Neural Network** este clar superiorul pentru probleme multi-class complexe
2. **Random Forest** bun pentru performanță generală dar ignoră clasele minoritare
3. **Custom Logistic Regression** demonstrează înțelegerea algoritmilor dar e limitată

Pentru Heart Dataset:

1. **Custom Logistic Regression** este cel mai potrivit
2. Acuratețea înaltă poate fi înșelătoare în cazuri medicale
3. **Recall-ul pentru clasa pozitivă** este metrica critică

Lecții Generale:

1. **Confusion matrices** dezvăluie mult mai mult decât accuracy-ul simplu
2. **Context aplicație** determină ce metrici sunt importante

3. **Ponderarea claselor** este crucială pentru date dezechilibrate
4. **Neural Networks** excel la probleme complexe cu date suficiente

6.5 Analiza Comparativă - Graficul de Performanță

Graficul de comparație evidențiază următoarele aspecte critice:

Ordinea Performanței (News Dataset):

1. **Neural Network (76.0%)**
2. **Random Forest (62.6%)**

3. **Decision Tree (56.5%)**
4. **Custom Logistic Regression (52.8%)**

De Ce Neural Network Domină:

- **Capacitate non-liniară**: Poate învăța patterns complexe între features
- **Regularizare automată**: Early stopping previne overfitting-ul
- **Optimizare adaptivă**: Adam optimizer se adaptează la gradienții din date
- **Arhitectură potrivită**: 100-50 neuroni suficient pentru complexitate fără overfitting

De Ce Random Forest E Solid dar Nu Excelent:

- **Bun la generalizare** dar nu cap turează relații fine între features
- **Bias către clasa majoritară** reduce versatilitatea
- **Ensemble ajută** dar algoritmul de bază rămâne simplu

De Ce Decision Tree E Mediocru:

- **Overfitting** în ciuda hiperparametrizării
- **Instabilitate** la mici schimbări în date
- **Incapacitate** de a captura relații complexe

De Ce Custom Logistic Regression E Ultimul în Accuracy:

- **Limitare liniară fundamentală** pe date non-liniare
- **Implementare simplă** fără regularizare avansată
- **Dar**: Cea mai bună la echilibrarea claselor!

Heart Dataset:

Deși Random Forest are accuracy-ul cel mai mare (84.91%), **Custom Logistic Regression (78.89%)** este practic cel mai util:

Metric	Random Forest	Custom Logistic
Accuracy	84.91%	78.89%
Recall clasa pozitivă	1.6%	32.6%
Pacienți cu risc detectați	2/129	42/129

6. Utilizare

Executarea completă:

```
python3 run_pipeline.py
```

Executarea individuală:

```
# Doar EDA
python3 data_explorer.py

# Doar preprocessing
python3 data_processor.py

# Doar training
python3 ml_orchestrator.py
```

7. Dependințe

```
pandas>=1.3.0
numpy>=1.21.0
matplotlib>=3.4.0
seaborn>=0.11.0
scikit-learn>=1.0.0
scipy>=1.7.0
```

8. Comentarii asupra Rezultatelor și Explicații de Performanță

8.1 De Ce Neural Network Obține Cea Mai Bună Performanță pe News Dataset

Factori Tehnici:

1. **Capacitate Non-Liniară Superioară**
   - Poate modela interacțiuni complexe între 52 de features
   - ReLU activation permite învățarea de pattern-uri sofisticate
   - Arhitectura 100-50 suficientă pentru complexitate fără overfitting
2. **Optimizare Adaptivă Eficientă**
   - Adam optimizer se adaptează la gradienții variabili din date
   - Learning rate adaptat pe dimensiuni diferite ale problemei

- Momentum ajută la navigarea prin space-uri complexe de parametri

3. **Regularizare Inteligentă**
   - Early stopping previne overfitting-ul automat
   - L2 regularization (alpha=0.001) balansează complexitatea
   - Dropout implicit în sklearn MLPClassifier
4. **Gestionarea Superioară a Clasei Dezechilibrate**
   - Reușește să învețe reprezentări distincte pentru fiecare clasă
   - Nu se limitează la clasificarea majoritară ca tree-based models

Dovezi din Metrici:

- **Singura cu recall > 15% pentru clasa Viral** (15% vs 1% pentru Random Forest)
- **Cel mai bun F1-score pentru 4/5 clase**
- **Precision echilibrată** între clase (0.45-0.81 vs 0.0-1.0 pentru Random Forest)

8.2 Alegerea Algoritmilor

Pentru Probleme Multi-Class Complexe (News): Neural Networks când: - Dataset mare (>30k exemple) - Features multe și diverse (>50) - Relații non-liniare suspectate - Classe dezechilibrate moderate

Pentru Probleme Medicale/Critice (Heart): Logistic Regression când: - Costul false negative »> false positive - Interpretabilitate necesară - Probabilități calibrate importante - Dataset moderat (<5k exemple)

Pentru Baseline Rapid: Random Forest când: - Timp de development limitat - Performance "decent" suficientă - Features mixed (numeric + categorical) - Robustețe la outliers necesară

8.3 Impactul Factorilor de Complexitate

Dimensiunea Dataset-ului:

- **News (31,715)**: Beneficiază de Neural Networks complexe
- **Heart (3,392)**: Susceptibil la overfitting cu modele complexe

Numărul de Features:

- **News (52 features)**: Necesită capacitate de modelare sofisticată
- **Heart (14 features)**: Modele simple pot fi suficiente

Dezechilibrul Claselor:

- **News**: 5 clase (2.7% - 47.9%) → Neural Network gestionează cel mai bine
- **Heart**: 2 clase (15.2% - 84.8%) → Ponderarea explicită în Logistic Regression decisivă