Risks

April 26, 2025

[ ]:
```
import pandas as pd

# Read your cleaned and saved abstracts dataset
df = pd.read_csv("/content/alzheimers_abstracts.csv")

# Define a list of risk factor keywords
risk_keywords = [
    "air pollution", "PM2.5", "particulate matter", "environmental exposure",
    "toxins", "neuroinflammation", "smoking", "hypertension", "cholesterol",
    "diet", "sleep quality", "obesity", "pesticides"
]

# Add a new column 'Factors': mark 1 if any keyword is found in the abstract, else 0
df["Factors"] = df["Abstract"].apply(
    lambda text: int(any(kw.lower() in text.lower() for kw in risk_keywords))
)

# Save the updated DataFrame to a new CSV file
df.to_csv("alzheimers_abstracts_risk.csv", index=False)

print("Successfully completed keyword tagging for risk factors and saved the new file.")
```

```
Successfully completed keyword tagging for risk factors and saved the new file.
```

[ ]:
```
from sklearn.model_selection import train_test_split

# Read your processed dataset (already labeled with 'Factors')
df = pd.read_csv("/content/alzheimers_abstracts_risk.csv")

# Split the dataset into Training set + Temporary set (dev + test)
train_df, temp_df = train_test_split(df, test_size=0.3, random_state=42, stratify=df["Factors"])

# Further split the Temporary set into Dev and Test sets (50% each)
dev_df, test_df = train_test_split(temp_df, test_size=0.5, random_state=42, stratify=temp_df["Factors"])
```

```
# Save the resulting datasets to new CSV files
train_df.to_csv("risk_train.csv", index=False)
dev_df.to_csv("risk_dev.csv", index=False)
test_df.to_csv("risk_test.csv", index=False)

print("Train/Dev/Test sets have been saved.")
```

```
Train/Dev/Test sets have been saved.
```

[ ]:
```
import pandas as pd
import spacy
from spacy.pipeline import EntityRuler

nlp = spacy.load("en_core_web_sm")

# Add an EntityRuler to insert custom entity recognition patterns before the␣
 ↪built-in NER
ruler = nlp.add_pipe("entity_ruler", before="ner")

# Define a list of risk factor keywords
risk_keywords = [
    "air pollution", "PM2.5", "particulate matter", "environmental exposure",
    "toxins", "neuroinflammation", "smoking", "hypertension", "cholesterol",
    "diet", "sleep quality", "obesity", "pesticides"
]

# Build custom patterns with label 'RISK'
patterns = [{"label": "RISK", "pattern": kw} for kw in risk_keywords]
ruler.add_patterns(patterns)

# Test entity recognition on a single simple sentence
doc = nlp("The patient experienced air pollution.")
print("\nTesting entity recognition on a sample sentence:")
for ent in doc.ents:
    print(f" - {ent.text} ({ent.label_})")

# Read the abstracts dataset
df = pd.read_csv("/content/alzheimers_abstracts.csv")

# Perform entity recognition on the first 5 abstracts
print("\nBatch processing entity recognition results:")
for i in range(5):
    text = df.loc[i, "Abstract"]
    print(f"\nAbstract #{i+1}:\n{text}")

    doc = nlp(text) # Apply NLP pipeline
```

```
    print("Recognized Entities:")
    for ent in doc.ents:
        print(f" - {ent.text} ({ent.label_})")
```

```
Testing entity recognition on a sample sentence:
 - air pollution (RISK)

Batch processing entity recognition results:

Abstract #1:
Whether or not neuropsychiatric symptoms (NPS) in advance of dementia are
associated with Alzheimer disease (AD) and/or other neurodegenerative dementias
remains to be determined. The mild behavioural impairment (MBI) construct
selects persons with NPS that are later-life emergent and persistent to identify
a high-risk group for cognitive decline and incident dementia. Here, in older
adults without dementia at baseline, we examined whether postmortem AD and other
neurodegenerative pathologies were associated with MBI in the five years before
death. National Alzheimer's Coordinating Center study autopsy participants
(n=1016, 82.6 years, 48.7% female, 60% normal cognition) were included in the
analyses. Using the Neuropsychiatric Inventory-Questionnaire, MBI+ status was
operationalized as NPS persistence at >2/3 of pre-dementia study visits;
otherwise, status was non-MBI NPS. The presence of AD, Lewy body disease (LBD),
and TDP-43 neuropathological changes were determined using published guidelines.
Adjusted multinomial logistic regressions modeled pathology-NPS status
associations. Adjusted Cox proportional hazards regressions modeled hazard for
AD-dementia at each NPS status level, including interaction terms with cognitive
status and each co-pathology. AD+ individuals (51.4%) were 88.4% more likely to
be MBI+  5 years prior than AD- individuals (odds ratio (OR):1.88, 95%
confidence interval (CI):1.29-2.75, p<0.01); however, the likelihood of having
non-MBI NPS was not different (OR:1.22, CI:0.90-1.66, p=0.20). No significant
associations were seen for LBD pathology, even among AD+ participants. There
were no significant differences in the levels of LBD or TDP-43 in those with MBI
compared to no MBI. Among MBI progressors to dementia (n=106), 33.0% were solely
AD+, 18.9% were mixed AD+/LBD+, and 11.3% had all three pathologies. For all
those with MBI (including dementia non-progressors), of persons with LBD, 83.4%
were comorbid with AD. In the survival analysis, MBI+ individuals had a
2.03-fold greater progression rate to AD-dementia than noNPS (CI: 1.60-2.57,
p<0.01). Progression rate was higher in MCI, but the effect of MBI on
progression was greater in NC (HR:3.05, CI:1.37-6.80, p<0.01) vs. MCI (HR:1.93,
CI:1.51-2.47, p<0.01). Limbic LBD appeared to also moderate the association
between MBI and incident AD (Limbic LBD+ HR: 4.64, CI: 2.05-10.50, p<0.001;
Limbic LBD- HR: 1.87, CI: 1.46-2.40, p<0.001). Antecedent MBI was strongly
associated with AD pathology but not with other neurodegenerative dementias.
Inclusion of MBI in research and clinical frameworks for dementia may aid in
identification of early stages of neurodegenerative disease, which may be
helpful for selecting patients for treatment with AD disease-modifying drugs.
```

Recognized Entities:
- NPS (ORG)
- NPS (ORG)
- the five years (DATE)
- National Alzheimer's (ORG)
- Coordinating Center (ORG)
- n=1016 (GPE)
- 82.6 years (DATE)
- 48.7% (PERCENT)
- 60% (PERCENT)
- the Neuropsychiatric Inventory-Questionnaire (FAC)
- NPS (ORG)
- 2/3 (CARDINAL)
- LBD (ORG)
- NPS (ORG)
- NPS (ORG)
- 51.4% (PERCENT)
- 88.4% (PERCENT)
- years (DATE)
- 95% (PERCENT)
- NPS (ORG)
- LBD (ORG)
- LBD (ORG)
- 33.0% (PERCENT)
- 18.9% (PERCENT)
- 11.3% (PERCENT)
- three (CARDINAL)
- LBD (ORG)
- 83.4% (PERCENT)
- 2.03-fold (CARDINAL)
- noNPS (PERSON)
- CI (ORG)
- 1.60-2.57 (CARDINAL)
- MCI (ORG)
- NC (GPE)
- HR:3.05 (ORG)
- CI:1.37-6.80 (ORG)
- MCI (ORG)
- Limbic LBD (ORG)
- 4.64 (CARDINAL)
- CI (ORG)
- 2.05 (CARDINAL)
- p<0.001 (ORG)
- Limbic LBD- (ORG)
- 1.87 (CARDINAL)
- CI (ORG)
- 1.46-2.40 (CARDINAL)

Abstract #2:
Neurodegenerative diseases, such as Alzheimer's and Parkinson's Disease, pose a significant healthcare burden to the aging population. Structural MRI brain parameters and accelerometry data from wearable devices have been proven to be useful predictors for these diseases but have been separately examined in the prior literature. This study aims to determine whether a combination of accelerometry data and MRI brain parameters may improve the detection and prognostication of Alzheimer's and Parkinson's disease, compared with MRI brain parameters alone. A cohort of 19,793 participants free of neurodegenerative disease at the time of imaging and accelerometry data capture from the UK Biobank with longitudinal follow-up was derived to test this hypothesis. Relevant structural MRI brain parameters, accelerometry data collected from wearable devices, standard polygenic risk scores and lifestyle information were obtained. Subsequent development of neurodegenerative diseases among participants was recorded (mean follow-up time of 5.9 years), with positive cases defined as those diagnosed at least one year after imaging. A machine learning algorithm (XGBoost) was employed to create prediction models for the development of neurodegenerative disease. A prediction model consisting of all factors, including structural MRI brain parameters, accelerometry data, PRS, and lifestyle information, achieved the highest AUC value (0.819) out of all tested models. A model that excluded MRI brain parameters achieved the lowest AUC value (0.688). Feature importance analyses revealed 18 out of 20 most important features were structural MRI brain parameters, while 2 were derived from accelerometry data. Our study demonstrates the potential utility of combining structural MRI brain parameters with accelerometry data from wearable devices to predict the incidence of neurodegenerative diseases. Future prospective studies across different populations should be conducted to confirm these study results and look for differences in predictive ability for various types of neurodegenerative diseases.
Recognized Entities:
- 19,793 (CARDINAL)
- UK (GPE)
- 5.9 years (DATE)
- at least one year (DATE)
- XGBoost (ORG)
- PRS (ORG)
- 0.819 (CARDINAL)
- 18 (CARDINAL)
- 20 (CARDINAL)
- 2 (CARDINAL)

Abstract #3:
Recent reports suggest dysregulation of the N6-methyladenosine (m6A) RNA modification may contribute to the pathology of neurodegenerative diseases. Herein, we show the m6A methyltransferase complex including METTL3-the catalytic component of the nuclear-localized complex-is robustly upregulated in human microglia and astrocytes exposed to Syn_f and Mn. Subcellular localization studies reveal METTL3 was predominantly cytoplasmic following Mn

insult but remained nuclear following Syn_f stimulation in activated microglia. Functional analysis revealed METTL3 and downstream m6A readers, including YTHDF2 and IGF2BP1-3, may regulate the proinflammatory secretome of activated microglia. Notably, methyltransferase activity and m6A abundance were significantly increased following Mn and Syn_f treatment. METTL3 in Mn and Syn_fin vivo models of neuroinflammation, along with human postmortem tissues from Alzheimer's disease (AD), Parkinson's disease (PD), and dementia with Lewy bodies (DLB) patients, was significantly upregulated. This was further confirmed by single-cell RNA sequencing (scRNA-seq) analysis. Overall, we demonstrate the m6A writer METTL3 may function as a major regulator of chronic neuroinflammation in synucleinopathies.
Recognized Entities:
- N6-methyladenosine (DATE)
- Herein (PERSON)
- METTL3 (PRODUCT)
- Syn (ORG)
- METTL3 (PRODUCT)
- Syn (ORG)
- METTL3 (PRODUCT)
- YTHDF2 (ORG)
- IGF2BP1-3 (DATE)
- Mn (ORG)
- Syn (ORG)
- Syn (ORG)
- neuroinflammation (RISK)
- Lewy (ORG)
- DLB (ORG)
- RNA (ORG)
- METTL3 (PRODUCT)
- neuroinflammation (RISK)

Abstract #4:
Aging is a slow and irreversible biological process leading to decreased cell and tissue functions with higher risks of multiple age-related diseases, including neurodegenerative diseases. It is widely accepted that aging represents the leading risk factor for neurodegeneration. The pathogenesis of these diseases involves complex interactions of genetic mutations, environmental factors, oxidative stress, neuroinflammation, and mitochondrial dysfunction, which complicate treatment with traditional mono-targeted therapies. Network pharmacology can help identify potential gene or protein targets related to neurodegenerative diseases. Integrating advanced molecular profiling technologies and computer-aided drug design further enhances the potential of network pharmacology, enabling the identification of biomarkers and therapeutic targets, thus paving the way for precision medicine in neurodegenerative diseases. This review article delves into the application of network pharmacology in understanding and treating neurodegenerative disorders such as Alzheimer's disease, Parkinson's disease, amyotrophic lateral sclerosis, Huntington's disease, and spinal muscular atrophy. Overall, this article

```
emphasizes the importance of addressing aging as a central factor in developing
effective disease-modifying therapies, highlighting how network pharmacology can
unravel the complex biological networks associated with aging and pave the way
for personalized medical strategies.
Recognized Entities:
 - neuroinflammation (RISK)
 - Huntington (ORG)

Abstract #5:
Alzheimer's disease (AD) is an age-related neurodegenerative disorder
characterized by insidious and gradual onset. Identifying biomarkers associated
with the early stages of AD is crucial for delaying its progression. In this
study, we aimed to identify AD-related biomarkers in blood and urine by
integrating genetic correlation analysis, shared genetic loci identification,
and causal inference using linkage disequilibrium score regression (LDSC),
conjunction false discovery rate (conjFDR), generalized summary data-based
Mendelian randomization (GSMR) and two-sample Mendelian randomization (MR). To
enhance robustness and minimize sample bias, we cross-validated findings using
different AD GWAS datasets. Across multiple AD GWASs, we consistently observed
nominally significant genetic correlations: AD was positively correlated with
albumin (ALB) and negatively correlated with cystatin C (CYS) and urea (BUN). MR
analysis further suggested that genetic predisposition to higher level of ALB
and lower level of non-albumin protein (NAP) can represent risk factors for AD.
In reverse MR analysis, a higher genetic risk for AD can predispose individuals
to higher levels of ratio of aspartate aminotransferase to alanine
aminotransferase (AST2ALT) and estimated glomerular filtration rate (EGFR), as
well as lower level of creatinine (CRE). Overall, this study provides insights
into the genetic correlations and causal relationships between AD and several
biomarkers, offering potential candidates for AD diagnosis and management.
Recognized Entities:
 - LDSC (PERSON)
 - Mendelian (NORP)
 - two (CARDINAL)
 - Mendelian (NORP)
 - ALB (ORG)
 - cystatin C (PERSON)
 - CYS (ORG)
 - BUN (ORG)
 - ALB (ORG)
 - NAP (ORG)
 - MR (GPE)
```

[ ]:
```python
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
import pandas as pd
```

```
# Load the training and testing datasets
train_df = pd.read_csv("/content/risk_train.csv")
test_df = pd.read_csv("/content/risk_test.csv")

# Extract features (text) and labels
X_train2 = train_df["Abstract"]
y_train2 = train_df["Factors"]
X_test2 = test_df["Abstract"]
y_test2 = test_df["Factors"]

# Apply TF-IDF vectorization to the text
vectorizer = TfidfVectorizer(max_features=5000)
X_train_vec2 = vectorizer.fit_transform(X_train2)
X_test_vec2 = vectorizer.transform(X_test2)

# Train a Logistic Regression model
clf = LogisticRegression(max_iter=200)
clf.fit(X_train_vec2, y_train2)

# Predict and print the classification report
y_pred2 = clf.predict(X_test_vec2)
print(classification_report(y_test2, y_pred2))
```

```
              precision    recall  f1-score   support

           0       0.89      1.00      0.94      1202
           1       0.97      0.36      0.53       240

    accuracy                           0.89      1442
   macro avg       0.93      0.68      0.73      1442
weighted avg       0.90      0.89      0.87      1442
```

[ ]:
```
from sklearn.metrics import classification_report, confusion_matrix,␣
↪roc_auc_score

# Predict on the test set
y_pred2 = clf.predict(X_test_vec2) # Predicted class labels (0 or 1)
y_prob2 = clf.predict_proba(X_test_vec2)[:, 1]

# Print a detailed classification report
print("Classification Report:")
print(classification_report(y_test2, y_pred2))

# Calculate and print ROC-AUC score
auc = roc_auc_score(y_test2, y_prob2)
print(f"ROC-AUC: {auc:.4f}")
```

```
from sklearn.metrics import confusion_matrix

# Compute and print the confusion matrix
cm = confusion_matrix(y_test2, y_pred2)
print("Confusion Matrix:")
print(cm)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.89      1.00      0.94      1202
           1       0.97      0.36      0.53       240

    accuracy                           0.89      1442
   macro avg       0.93      0.68      0.73      1442
weighted avg       0.90      0.89      0.87      1442

ROC-AUC: 0.9456
Confusion Matrix:
[[1199    3]
 [ 153   87]]
```

[ ]:
```
! pip install datasets
```

```
Collecting datasets
  Downloading datasets-3.5.0-py3-none-any.whl.metadata (19 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-
packages (from datasets) (3.18.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-
packages (from datasets) (2.0.2)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages
(from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in
/usr/local/lib/python3.11/dist-packages (from datasets) (2.32.3)
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.11/dist-
packages (from datasets) (4.67.1)
Collecting xxhash (from datasets)
  Downloading
xxhash-3.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(12 kB)
Collecting multiprocess<0.70.17 (from datasets)
  Downloading multiprocess-0.70.16-py311-none-any.whl.metadata (7.2 kB)
Collecting fsspec<=2024.12.0,>=2023.1.0 (from
```

```
fsspec[http]<=2024.12.0,>=2023.1.0->datasets)
  Downloading fsspec-2024.12.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-
packages (from datasets) (3.11.15)
Requirement already satisfied: huggingface-hub>=0.24.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (0.30.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-
packages (from datasets) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-
packages (from datasets) (6.0.2)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-
packages (from aiohttp->datasets) (25.3.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.6.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (6.4.3)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (0.3.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.20.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.24.0->datasets)
(4.13.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets)
(3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-
packages (from requests>=2.32.2->datasets) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets)
(2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets)
(2025.1.31)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas->datasets) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas->datasets) (2025.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-
packages (from python-dateutil>=2.8.2->pandas->datasets) (1.17.0)
Downloading datasets-3.5.0-py3-none-any.whl (491 kB)
                           491.2/491.2 kB
```

```
33.5 MB/s eta 0:00:00
Downloading dill-0.3.8-py3-none-any.whl (116 kB)
                        116.3/116.3 kB
12.6 MB/s eta 0:00:00
Downloading fsspec-2024.12.0-py3-none-any.whl (183 kB)
                        183.9/183.9 kB
19.8 MB/s eta 0:00:00
Downloading multiprocess-0.70.16-py311-none-any.whl (143 kB)
                        143.5/143.5 kB
15.8 MB/s eta 0:00:00
Downloading
xxhash-3.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
                        194.8/194.8 kB
20.9 MB/s eta 0:00:00
Installing collected packages: xxhash, fsspec, dill, multiprocess,
datasets
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2025.3.2
    Uninstalling fsspec-2025.3.2:
      Successfully uninstalled fsspec-2025.3.2
```

```
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
gcsfs 2025.3.2 requires fsspec==2025.3.2, but you have fsspec 2024.12.0 which is
incompatible.
torch 2.6.0+cu124 requires nvidia-cublas-cu12==12.4.5.8; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cublas-cu12
12.5.3.2 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-cupti-cu12==12.4.127; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cuda-cupti-cu12
12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-nvrtc-cu12==12.4.127; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cuda-nvrtc-cu12
12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-runtime-cu12==12.4.127; platform_system
== "Linux" and platform_machine == "x86_64", but you have nvidia-cuda-runtime-
cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cudnn-cu12==9.1.0.70; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cudnn-cu12
9.3.0.75 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cufft-cu12==11.2.1.3; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cufft-cu12
11.2.3.61 which is incompatible.
torch 2.6.0+cu124 requires nvidia-curand-cu12==10.3.5.147; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-curand-cu12
10.3.6.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusolver-cu12==11.6.1.9; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cusolver-cu12
11.6.3.83 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusparse-cu12==12.3.1.170; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cusparse-cu12
12.5.1.3 which is incompatible.
torch 2.6.0+cu124 requires nvidia-nvjitlink-cu12==12.4.127; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-nvjitlink-cu12
12.5.82 which is incompatible.
Successfully installed datasets-3.5.0 dill-0.3.8 fsspec-2024.12.0
```

```
multiprocess-0.70.16 xxhash-3.5.0
```

[ ]:
```python
from transformers import DistilBertTokenizerFast,␣
 ↪DistilBertForSequenceClassification, Trainer, TrainingArguments
from datasets import Dataset
import pandas as pd
import numpy as np
from sklearn.metrics import classification_report, roc_auc_score

# Load the train and test datasets
df = pd.read_csv("/content/risk_train.csv")
df_test = pd.read_csv("/content/risk_test.csv")

# Load the DistilBERT tokenizer
tokenizer = DistilBertTokenizerFast.from_pretrained("distilbert-base-uncased")

# Preprocessing: prepare Huggingface Dataset objects
# Rename columns to match Huggingface expected names ("text", "label")
train_dataset2 = Dataset.from_pandas(df[["Abstract", "Factors"]].
 ↪rename(columns={"Abstract": "text", "Factors": "label"}))
test_dataset2 = Dataset.from_pandas(df_test[["Abstract", "Factors"]].
 ↪rename(columns={"Abstract": "text", "Factors": "label"}))
# Apply tokenization
train_dataset2 = train_dataset2.map(lambda x: tokenizer(x["text"],␣
 ↪truncation=True, padding="max_length"), batched=True)
test_dataset2 = test_dataset2.map(lambda x: tokenizer(x["text"],␣
 ↪truncation=True, padding="max_length"), batched=True)

# Define label map and load model
id2label = {0: "No risk", 1: "Mentions risk"}
label2id = {"No risk": 0, "Mentions risk": 1}

# load model
from transformers import AutoModelForSequenceClassification

model = AutoModelForSequenceClassification.from_pretrained(
    "distilbert-base-uncased",
    num_labels=2,
    id2label=id2label,
    label2id=label2id
)


# Set training parameters
from transformers import TrainingArguments
```

```
from transformers import (
    AutoTokenizer, DataCollatorWithPadding, AutoModelForSequenceClassification,
    TrainingArguments, Trainer
)
training_args = TrainingArguments(
    output_dir="text_classification_model",
    learning_rate=2e-5,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=4,
    weight_decay=0.01,
    eval_strategy="epoch",                        # run eval at the end of each␣
 ↪epoch
    save_strategy="epoch",
    load_best_model_at_end=True,
    push_to_hub=False,
    report_to="none",
    fp16=True  # using a GPU with FP16 support with Colab
)


# Define custom evaluation metrics
def compute_metrics(pred):
    labels = pred.label_ids
    preds = np.argmax(pred.predictions, axis=1)
    auc = roc_auc_score(labels, pred.predictions[:, 1])
    report = classification_report(labels, preds, output_dict=True)
    return {
        "accuracy": report["accuracy"],
        "precision": report["1"]["precision"],
        "recall": report["1"]["recall"],
        "f1": report["1"]["f1-score"],
        "roc_auc": auc
    }

# Initialize Trainer and train model
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset2,
    eval_dataset=test_dataset2,
    compute_metrics=compute_metrics
)

trainer.train() # Start model training
trainer.evaluate() # Evaluate model on the test set
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94:
```

```
UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab
(https://huggingface.co/settings/tokens), set it as secret in your Google Colab
and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access
public models or datasets.
  warnings.warn(

tokenizer_config.json:   0%|          | 0.00/48.0 [00:00<?, ?B/s]

vocab.txt:   0%|          | 0.00/232k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/466k [00:00<?, ?B/s]

config.json:   0%|          | 0.00/483 [00:00<?, ?B/s]

Map:   0%|          | 0/6727 [00:00<?, ? examples/s]

Map:   0%|          | 0/1442 [00:00<?, ? examples/s]

Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed.
Falling back to regular HTTP download. For better performance, install the
package with: `pip install huggingface_hub[hf_xet]` or `pip install hf_xet`
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but
the 'hf_xet' package is not installed. Falling back to regular HTTP download.
For better performance, install the package with: `pip install
huggingface_hub[hf_xet]` or `pip install hf_xet`

model.safetensors:   0%|          | 0.00/268M [00:00<?, ?B/s]

Some weights of DistilBertForSequenceClassification were not initialized from
the model checkpoint at distilbert-base-uncased and are newly initialized:
['classifier.bias', 'classifier.weight', 'pre_classifier.bias',
'pre_classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

[ ]:
```
{'eval_loss': 0.03623471036553383,
 'eval_accuracy': 0.9937586685159501,
 'eval_precision': 0.9957081545064378,
 'eval_recall': 0.9666666666666667,
 'eval_f1': 0.9809725158562368,
 'eval_roc_auc': 0.9954468247365502,
 'eval_runtime': 5.7688,
 'eval_samples_per_second': 249.964,
 'eval_steps_per_second': 15.774,
 'epoch': 4.0}
```

[ ]:
```
import transformers
print(transformers.__version__)
```

```
4.51.3
```

[ ]:
```
! pip install evaluate
```

```
Collecting evaluate
  Downloading evaluate-0.4.3-py3-none-any.whl.metadata (9.2 kB)
Requirement already satisfied: datasets>=2.0.0 in
/usr/local/lib/python3.11/dist-packages (from evaluate) (3.5.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-
packages (from evaluate) (2.0.2)
Requirement already satisfied: dill in /usr/local/lib/python3.11/dist-packages
(from evaluate) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages
(from evaluate) (2.2.2)
Requirement already satisfied: requests>=2.19.0 in
/usr/local/lib/python3.11/dist-packages (from evaluate) (2.32.3)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.11/dist-
packages (from evaluate) (4.67.1)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages
(from evaluate) (3.5.0)
Requirement already satisfied: multiprocess in /usr/local/lib/python3.11/dist-
packages (from evaluate) (0.70.16)
Requirement already satisfied: fsspec>=2021.05.0 in
/usr/local/lib/python3.11/dist-packages (from fsspec[http]>=2021.05.0->evaluate)
(2024.12.0)
Requirement already satisfied: huggingface-hub>=0.7.0 in
/usr/local/lib/python3.11/dist-packages (from evaluate) (0.30.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-
packages (from evaluate) (24.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-
packages (from datasets>=2.0.0->evaluate) (3.18.0)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from datasets>=2.0.0->evaluate)
(18.1.0)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-
packages (from datasets>=2.0.0->evaluate) (3.11.15)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-
packages (from datasets>=2.0.0->evaluate) (6.0.2)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.7.0->evaluate)
(4.13.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->evaluate)
(3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-
```

```
packages (from requests>=2.19.0->evaluate) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->evaluate)
(2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->evaluate)
(2025.1.31)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas->evaluate) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas->evaluate) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas->evaluate) (2025.2)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-
packages (from aiohttp->datasets>=2.0.0->evaluate) (25.3.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (1.6.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (6.4.3)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (0.3.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (1.20.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-
packages (from python-dateutil>=2.8.2->pandas->evaluate) (1.17.0)
Downloading evaluate-0.4.3-py3-none-any.whl (84 kB)
                         84.0/84.0 kB
8.9 MB/s eta 0:00:00
Installing collected packages: evaluate
Successfully installed evaluate-0.4.3
```

[ ]:
```python
# Define evaluation metric
import evaluate
accuracy = evaluate.load("accuracy")
```

```
Downloading builder script:   0%|          | 0.00/4.20k [00:00<?, ?B/s]
```

[ ]:
```python
# Custom function to compute metrics during evaluation
def compute_metrics(eval_pred):
    predictions2, labels = eval_pred
    preds2 = np.argmax(predictions2, axis=1)
    return accuracy.compute(predictions=preds2, references=labels)
```

[ ]:
```python
# Make predictions on the test dataset using the trained model
predictions2 = trainer.predict(test_dataset2)
```

```
<IPython.core.display.HTML object>
```

[ ]:
```python
# Extract predicted labels and true labels
y_pred2 = np.argmax(predictions2.predictions, axis=1)
y_true2 = predictions2.label_ids
# Define evaluation metric
import evaluate
accuracy = evaluate.load("accuracy")
```

[ ]:
```python
# Print detailed classification report
print("\nClassification Report:")
report = classification_report(
    y_true2, y_pred2,
    labels=[0, 1], # Define the order of labels
    target_names=["No risk", "Mentions risk"], # Define label names
    digits=4 # Display results with 4 decimal places
)
print(report)

from sklearn.metrics import confusion_matrix, accuracy_score
# Print confusion matrix and overall accuracy
print("\nConfusion Matrix:")
print(confusion_matrix(y_true2, y_pred2, labels=[0, 1]))

print("\nOverall Accuracy:", accuracy_score(y_true2, y_pred2))
```

```
Classification Report:
              precision    recall  f1-score   support

      No risk     0.9934    0.9992    0.9963      1202
Mentions risk     0.9957    0.9667    0.9810       240

     accuracy                         0.9938      1442
    macro avg     0.9945    0.9829    0.9886      1442
 weighted avg     0.9938    0.9938    0.9937      1442


Confusion Matrix:
```

```
[[1201    1]
 [   8  232]]

Overall Accuracy: 0.9937586685159501
```