**Mistral**

**1 - First Prompt:**

def llm\_predict(row):

prompt = f"""\*\*Task\*\*: Predict the 10-year coronary heart disease risk using the following patient data.

\*\*Patient Data\*\*:

- Gender (male): {'Male' if row['male'] == 1 else 'Female'}

- Age (age): {row['age']} years

- Education (education): {row['education']} (1: Less than high school, 2: High school, 3: College, 4: Postgraduate)

- Smoking status (currentSmoker): {'Current smoker' if row['currentSmoker'] == 1 else 'Non-smoker'}

- Cigarettes per day (cigsPerDay): {row['cigsPerDay']}

- BP medication (BPMeds): {'On medication' if row['BPMeds'] == 1 else 'No medication'}

- Total cholesterol (totChol): {row['totChol']} mg/dL

- Systolic BP (sysBP): {row['sysBP']} mmHg

- Diastolic BP (diaBP): {row['diaBP']} mmHg

- Body mass index (BMI): {row['BMI']}

- Heart rate (heartRate): {row['heartRate']} bpm

- Fasting glucose (glucose): {row['glucose']} mg/dL

\*\*Clinical Criteria\*\* (Framingham Risk Score):

1. Age > 50 years: +risk

2. Systolic BP > 140 mmHg: +risk

3. Total cholesterol > 240 mg/dL: +risk

4. BMI > 30: +risk

5. Current smoker: +risk

6. Diabetes present: +risk

7. History of hypertension: +risk

\*\*Instructions\*\*:

1. Analyze using the above criteria

2. Evaluate each risk factor systematically

3. Return only 0 or 1 as specified

\*\*Output Format\*\*:

0: Low risk (<15% 10-year risk)

1: High risk (≥15% 10-year risk)

\*\*Your prediction\*\*:"""

response = llm.invoke([

{"role": "system", "content": "You are a cardiology specialist assessing 10-year coronary heart disease risk."},

{"role": "user", "content": prompt}

])

output = response.content.strip()

# Improved output parsing

try:

return int(float(output)) # Handles cases where output might be "1.0"

except ValueError:

return int('1' in output) # Fallback to string check if conversion fails

17m 54s execution time - 0.2 accuracy

**2 - Second:**

def llm\_predict(row):

prompt = f"""Analyze this patient's CHD risk (10 years). Return only 0 (low risk) or 1 (high risk).

Key Factors:

Age:{row['age']} | BP:{row['sysBP']}/{row['diaBP']} | Chol:{row['totChol']}

Smoker:{row['currentSmoker']} | BMI:{row['BMI']:.1f} | Diabetes:{row['diabetes']}

Thresholds:

- High risk if: Age>50 + BP>140/90 + Chol>240

- Or any 2 major: Smoking, BMI>30, Diabetes

Prediction (0/1):"""

response = llm.invoke([

{"role": "system", "content": "You're a cardiology AI. Assess CHD risk using only the provided data."},

{"role": "user", "content": prompt}

])

# Fast output parsing

output = response.content.strip()

return 1 if any(k in output for k in ['1','high','yes']) else 0

2m 57s - 0.60 accuracy / for 20 samples

4m 30s - 0.48 accuracy / for 25 samples

1m 19s - 0.60 accuracy / for 10 samples

2m 27s - 0.50 accuracy / for 10 samples

1m 55s - 0.50 accuracy / for 10 samples

1m 1s - 0.70 accuracy / for 10 samples

-------------------------------------------------------------------------------------------------------------------------

With JSON:

def enhanced\_predict(row):

*# Extract and format features*

features = {

'age': int(row['age']),

'bp': f"{int(row['sysBP'])}/{int(row['diaBP'])}",

'chol': int(row['totChol']),

'smoker': bool(row['currentSmoker']),

'bmi': float(row['BMI']),

'diabetes': bool(row['diabetes'])

}

prompt = f"""As a cardiology specialist, evaluate this patient's 10-year CHD risk:

Patient Data:

- Age: {features['age']}

- Blood Pressure: {features['bp']} mmHg

- Cholesterol: {features['chol']} mg/dL

- Smoker: {'Yes' if features['smoker'] else 'No'}

- BMI: {features['bmi']:.1f}

- Diabetes: {'Yes' if features['diabetes'] else 'No'}

Analysis Guidelines:

1. Calculate Framingham Risk Score:

- Age >50 (1 point)

- SBP >140 or DBP >90 (1 point)

- Cholesterol >240 (1 point)

- Smoking (1 point)

- BMI >30 (1 point)

- Diabetes (2 points)

2. Score ≥3 indicates high risk

Decision:

- Only respond with '0' (low risk) or '1' (high risk)

- No explanations needed

Your prediction:"""

try:

response = llm.invoke([

{"role": "system", "content": "You are a cardiac risk assessment AI. Follow the instructions precisely."},

{"role": "user", "content": prompt}

])

return int(response.content.strip()[:1]) *# Takes first character only*

except:

return 0 *# Fallback to low risk on error*

5m 23s - 0.00 accuracy / 20 samples

22.5s - 0.45 accuracy / 20 samples

16.9s - 0.50 accuracy / 20 samples

17.4s - 0.45 accuracy / 20 samples

def enhanced\_predict(row):

# Convert numeric fields safely

age = float(row['age']) if pd.notna(row['age']) else 0

sysBP = float(row['sysBP']) if pd.notna(row['sysBP']) else 0

diaBP = float(row['diaBP']) if pd.notna(row['diaBP']) else 0

totChol = float(row['totChol']) if pd.notna(row['totChol']) else 0

bmi = float(row['BMI']) if pd.notna(row['BMI']) else 0

prompt = f"""Analyze CHD risk (10 years). Return only 0 (low) or 1 (high).

Key Factors:

Age:{age:.0f} | BP:{sysBP:.0f}/{diaBP:.0f} | Chol:{totChol:.0f}

Smoker:{row['currentSmoker']} | BMI:{bmi:.1f} | Diabetes:{row['diabetes']}

Thresholds:

- High risk if: Age>50 + BP>140/90 + Chol>240

- Or any 2 major: Smoking, BMI>30, Diabetes

Prediction (0/1):"""

response = llm.invoke([

{"role": "system", "content": "You're a cardiology AI. Assess CHD risk using only the provided data."},

{"role": "user", "content": prompt}

])

output = response.content.strip()

return 1 if any(k in output for k in ['1','high','yes']) else 0

3m 55s - 0.25 accuracy / 20 samples

------------------------------------------------------------------------------------------------------------------------

One Shot:

def predict\_with\_one\_shot(row, example):

"""Make prediction using one-shot learning"""

prompt = f"""As a cardiology AI, assess 10-year CHD risk using this example:

{example}

Now evaluate this new patient:

- Age: {int(row['age'])}

- BP: {row['sysBP']:.0f}/{row['diaBP']:.0f}

- Cholesterol: {row['totChol']:.0f}

- Smoker: {'Yes' if row['currentSmoker'] else 'No'}

- BMI: {row['BMI']:.1f}

- Diabetes: {'Yes' if row['diabetes'] else 'No'}

Instructions:

1. Compare to the example

2. Analyze risk factors

3. Return ONLY 0 (low risk) or 1 (high risk)

Your prediction:"""

response = llm.invoke([

{"role": "system", "content": "You are a cardiac risk assessment AI."},

{"role": "user", "content": prompt}

])

# Robust output parsing

output = response.content.strip()

if output.startswith(('0', '1')):

return int(output[0])

return 0 # Default to low risk if unclear

2m 59s - 0.85 accuracy / 20 samples

3m 25s - 0.80 accuracy / 20 samples