Key factors (features) that can indicate a higher likelihood of betrayal.

Wealth related factors

- (Soldier's net wealth – Average wealth of soldiers) [Numerical Data]; If the soldier's net wealth is much more than the average wealth, there might be chances of corruption.
- Family Needs [Numerical Data]; If the soldier has a lot of members dependent on him in his family or if the family need has increased recently due to any natural or human calamities, he might get corrupted to meet his ends.

Power related factors

- (Increments in salaries and ranks per unit duration of service) [Numerical Data]; If the soldier is dissatisfied with his ranks or salaries over the years, he may switch sides for it.

Background related factors

- Number of members in the soldier's family who are serving in the army or have served in the past [Numerical Data]; A soldier belonging to a family of veterans or martyrs must have had a more patriotic upbringing.
- Does the soldier have any cultural relations (religious or linguistic) with the Phrygians [Categorical: Yes/No]; His loyalty towards the people of his religion or ethnicity might be more than his loyalty towards the kingdom.

Other factors

- Number of wars/operations, the soldier has been to [Numerical Data]; If the soldier has been to a lot wars previously, he is loyal to the kingdom.
- Relations with other mates [Categorical Data: Good/Average/Bad]. If the soldier's relations with his mates is poor, probabilities of betrayal are higher.
- Has the soldier ever been involved in any internal conflicts or revolutions [Categorical: Yes/No]; If yes, then there are chances of a coup.
- Number of weeks the soldier was posted near the Phrygian territories [Numerical Data]; If the soldier has had more exposure to the Phrygians, he has more chances of joining them.

The above features can be determined form the soldiers' personal records, military records, socio-economic records etc.

Other than that, if we (The Xernians) have a reliable and efficient intelligence, which has sufficient data about the soldiers, we can also include the following features:

- Number of times the soldier has visited the Phrygian territories [Numerical Data]
- Has the soldier ever had any personal interactions or relations with the Phrygians [Categorical Data: Yes/No]

If a soldier is truly loyal to his kingdom, the loyalty will prevail irrespective of all the social, economic and family factors and vice-versa. Hence, the most important factors in this scenario are the psychological factors. These are difficult to assess quantitatively as they are abstract.

But if through psychological assessments and reviews, we can rate them on the basis of:

- Ethical values: Soldiers with better ethical values are less likely to betray.
- Materialism: Soldiers attached to materialistic things is probably greedier and can betray for wealth.
- Risk-taking ability: Soldiers with lower risk-taking abilities might switch sides for their safety if they feel that the Phrygians are more powerful.
- Emotional stability: Soldiers prone to emotional volatility may be more likely to make impulsive decisions.
- Desire for autonomy: Soldiers who feel constrained by rigid hierarchies might try to free themselves form it.

All these factors can be rated on a scale of 1-10.

Designing a Decision-Making System:

The core of the system would be a risk score calculating algorithm that assigns each soldier a betrayal likelihood/risk score in a range (can be from 0-1 or 0-10 etc.) based on the weighted sum of the features.

- Each factor would be weighted according to its importance (determined through historical training data or expert judgement).
- A classification algorithm (e.g., XGBoost, Random Forest, Logistic Regression,) can be trained using historical data of soldiers who have previously defected.

- The system would learn from past betrayals and adjust weights of different factors to improve prediction accuracy as more data is collected.

Workflow of the system

Step 1: Data Collection

- The system continually gathers data from the social records, economic records, personal records, military records, intelligence reports and psychological assessments/reviews.
- We can process this data using the **pandas** and **numpy.**

Step 2: Feature Extraction and Pre-Processing

Transform raw data into usable features:

First of all, we will be converting all categorical features into numerical values by using **one hot encoding** and **ordinal encoding**. We can use the **Scikit Learn** library for that.

- Variables where relative order has a meaning, there ordinal will be used and where all categories are equivalent there, we will be using one hot encoding

Then we will be handling the missing values using imputation:

- **Scikit Learn** (from sklearn.impute import SimpleImputer) **or pandas** (for simple mean,median filling) again depending on the number and pattern of missing values.

Step 3: Risk Prediction Model

- We will be using Random Forest Classifier for our model. XGBoost can also be used for even better results.
- We will be using cross validation to tune our hyperparameters

Using this a **risk score** will be calculated for all soldiers and soldiers above a certain pre decided threshold will be marked 'risky'.

Model Evolution Using Fresh Data

To make a model such that it can evolve and improve as new data is added, we will implement libraries that allow this feature

Techniques for Evolving the System:

  - We will use algorithms like **partial_fit()** in Scikit-learn to update the model incrementally as new data is collected.

- **Feedback:**
  - We will create a system that collects feedback continuously (e.g., results of recent betrayals or loyalty confirmation from soldiers with high-risk scores).
  - Apart from updates that take place when a certain event related to betrayal happens, updates will also take place at regular fixed intervals to process new data received under various features.
  - In this case we would be updating the data after surveys or incidents of betrayal. So, we will be using batch learning.

Full Stack Design

Here's a proposed stack for implementing this evolving system:

- **Languages:**
  - **Python:** fundamental language for any ML related task.
- **Libraries:**

    - **Pandas, NumPy, Scikit-learn:** For data pre processing
    - **XGBoost or LightGBM:** Machine learning models' libraries.
    - **Model Update: Scikit-learn's partial_fit** for online learning (if necessary) or retraining periodically based on feedback.

- **Deployment:**
    - **Flask/Django API:** Wrap the model into a web service for making predictions.
    - **AWS/GCP for deployment and scaling.**