COMPLETE STRUCTURE FOR A ROBUST ML MODEL

This document outlines a comprehensive workflow for building and evaluating a machine learning model, followed by a summary of common ML models and their applications.

1. IMPORT REQUIRED LIBRARIES

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split,
GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,
classification_report
import joblib
```

2. LOAD AND EXPLORE DATA

```
df = pd.read_csv('data.csv')
print(df.info())
print(df.describe())
```

Tasks:
Check **data types**
Identify **missing values**
Understand **basic statistics**

3. DATA PREPROCESSING

Handle **missing values** (drop or impute)
Encode **categorical variables**
Remove **outliers**
Feature scaling (if needed)

4. SPLIT FEATURES AND TARGET

```
X = df.drop(columns=['target'])
y = df['target']
```

5. TRAIN-TEST SPLIT

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

6. FEATURE SCALING *(FOR MODELS THAT NEED SCALING)*

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

7. BASE MODEL TRAINING *(EXAMPLE: RANDOM FOREST)*

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

8. HYPERPARAMETER TUNING

```
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10]
}

grid_search =
GridSearchCV(RandomForestClassifier(random_state=42),
param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

best_model = grid_search.best_estimator_
```

9. EVALUATE MODEL

```
y_pred = best_model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

10. SAVE BEST MODEL

```
joblib.dump(best_model, 'best_model.pkl')
```

🏆 COMMON MACHINE LEARNING MODELS AND THEIR USES

✅ 1. LINEAR REGRESSION

Type: Regression
Use: Predict continuous numeric values
Example: Predicting house rent based on size, BHK, location
Advantages: Simple, interpretable
Disadvantages: Assumes linearity

✅ 2. LOGISTIC REGRESSION

Type: Classification
Use: Binary or multi-class prediction
Example: Will the tenant pay rent on time (Yes/No)?
Advantages: Probabilistic output
Disadvantages: Not good for non-linear data

✅ 3. DECISION TREE

Type: Both (Regression & Classification)
Use: Handles non-linear relationships
Example: Classify property as luxury or standard
Advantages: Easy to interpret
Disadvantages: Prone to overfitting

✅ 4. RANDOM FOREST

Type: Both
Use: Ensemble of decision trees
Example: Predict house rent with many features
Advantages: High accuracy
Disadvantages: Slower than single tree

✅ 5. GRADIENT BOOSTING / XGBOOST

Type: Both
Use: High-performance predictive modeling
Example: Predict real estate prices accurately
Advantages: Handles missing values
Disadvantages: More complex to tune

✅ 6. K-NEAREST NEIGHBORS (KNN)

Type: Both
Use: Similarity-based predictions
Example: Recommend rent based on similar properties
Advantages: Simple
Disadvantages: Slow for large datasets

✅ 7. SUPPORT VECTOR MACHINE (SVM)

Type: Both
Use: High-dimensional classification
Example: Classify tenants into risk categories
Advantages: Effective in high dimensions
Disadvantages: Slow on large datasets

✅ 8. NAIVE BAYES

Type: Classification
Use: Text classification, spam detection
Example: Classify tenant reviews as positive/negative
Advantages: Fast, simple
Disadvantages: Assumes independence of features

✅ 9. NEURAL NETWORKS (ANN)

Type: Both
Use: Complex patterns, deep learning
Example: Predict property price from multiple features
Advantages: Handles complex relationships
Disadvantages: Needs more data and compute

✅ 10. CLUSTERING (K-MEANS, DBSCAN)

Type: Unsupervised
Use: Group similar data points
Example: Cluster properties into low, medium, high rent segments
Advantages: No labeled data needed
Disadvantages: Needs tuning of number of clusters

✅ 11. PRINCIPAL COMPONENT ANALYSIS (PCA)

Type: Dimensionality Reduction
Use: Reduce number of features while keeping most variance
Example: Reduce correlated features in housing dataset
Advantages: Removes redundancy
Disadvantages: Harder to interpret transformed features