**Projeto: Análise e Previsão de Sorteios da Lotofácil**

**1. Objetivo**

Criar um sistema capaz de **ler arquivos CSV/XLSX** com históricos de sorteios da Lotofácil, realizar **análises estatísticas e predições** baseadas em inteligência artificial e simulações matemáticas para sugerir combinações otimizadas para futuras apostas.

**2. Estrutura do Projeto**

📂 lotofacil-analise  
┣ 📜 analise\_lotofacil.py → Código principal  
┣ 📜 dados.py → Funções para carregar e limpar dados  
┣ 📜 estatisticas.py → Análises estatísticas dos sorteios  
┣ 📜 predicao.py → Modelagem preditiva e Machine Learning  
┣ 📜 gerador\_jogos.py → Geração dos jogos recomendados  
┣ 📂 data/ → Pasta onde o arquivo **Lotofácil.xlsx** é armazenado

**3. Configuração do Ambiente**

Antes de rodar o projeto, instale as bibliotecas necessárias usando:

pip install pandas numpy matplotlib seaborn scikit-learn

**4. Carregamento e Preparo dos Dados**

📜 dados.py

import pandas as pd def carregar\_dados(arquivo="data/Lotofácil.xlsx"): """ Carrega os dados do arquivo da Lotofácil """ df = pd.read\_excel(arquivo) colunas\_numeros = [f"Bola{i}" for i in range(1, 16)] df\_numeros = df[colunas\_numeros] return df\_numeros

**5. Estatísticas Básicas**

📜 estatisticas.py

def analisar\_frequencia(df): """ Calcula a frequência de cada número sorteado """ frequencia = df.melt(value\_name="Numero")["Numero"].value\_counts() return frequencia def verificar\_padroes(df): """ Identifica repetições e padrões comuns nos sorteios """ sequencias = df.apply(lambda row: tuple(sorted(row)), axis=1) padroes\_comuns = sequencias.value\_counts() return padroes\_comuns.head(10)

**6. Modelagem e Predição**

📜 predicao.py

from sklearn.model\_selection import train\_test\_split from sklearn.ensemble import RandomForestClassifier import numpy as np import pandas as pd def treinar\_modelo(df): X = df.shift(1).dropna() y = df.iloc[X.index] # Garante que y tenha a mesma dimensão de X X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) modelo = RandomForestClassifier() modelo.fit(X\_train, y\_train) return modelo def simulacao\_monte\_carlo(modelo, n\_simulacoes=1000): resultados = [tuple(sorted(np.random.choice(range(1, 26), 15, replace=False))) for \_ in range(n\_simulacoes)] frequencias = pd.Series(resultados).value\_counts() return frequencias.head(10)



**7. Geração de Jogos**

📜 gerador\_jogos.py

from predicao import simulacao\_monte\_carlo def gerar\_jogos(df, modelo, quantidade=10): """ Gera um conjunto de jogos baseado nas probabilidades aprendidas """ melhores\_combinacoes = simulacao\_monte\_carlo(modelo, 5000) jogos\_sugeridos = melhores\_combinacoes.head(quantidade).index return jogos\_sugeridos

**8. Código Principal**

📜 analise\_lotofacil.py

from dados import carregar\_dados from estatisticas import analisar\_frequencia, verificar\_padroes from predicao import treinar\_modelo, simulacao\_monte\_carlo from gerador\_jogos import gerar\_jogos df = carregar\_dados() frequencia = analisar\_frequencia(df) print("📊 Frequência dos números sorteados:\n", frequencia.head(10)) padroes = verificar\_padroes(df) print("🔍 Padrões de sorteios recorrentes:\n", padroes.head(5)) modelo = treinar\_modelo(df) simulacao = simulacao\_monte\_carlo(modelo) print("🎲 Simulação de Monte Carlo - combinações prováveis:\n", simulacao.head(10)) jogos = gerar\_jogos(df, modelo, quantidade=5) print("🎯 Sugestões de apostas:\n", jogos)



**9. Como Executar**

1. Certifique-se de que **Lotofácil.xlsx** está na pasta data/.
2. **Rode cada módulo separadamente** para testar:

python dados.py python estatisticas.py python predicao.py python gerador\_jogos.py

1. Depois, execute **o código principal** para ver os resultados:

python analise\_lotofacil.py

**10. Melhorias Futuras**

🔹 **Gráficos interativos** para visualizar padrões  
🔹 **Aprimorar machine learning** com redes neurais  
🔹 **Automatizar coleta de novos sorteios online**

💾 **Agora é só copiar esse conteúdo para um documento Word ou Google Docs e exportar como PDF!**  
Me avise se quiser adicionar mais detalhes! 🚀