AJEENKYA
D Y PATIL UNIVERSITY
THE INNOVATION UNIVERSITY

A
MINI PROJECT REPORT ON

“ MLBB Hero Recommender ”

FOR

Term Work Examination

Bachelor of Computer Application in Artificial Intelligence and Machine Learning(BCA- AIML)

Year 2024-2025

Ajeenkya DY Patil University, Pune

-Submitted By-

Mr. Samyak Shinde

Under the guidance of

Prof. Vivek More

Ajeenkya DY Patil University

D Y Patil Knowledge City,
Charholi Bk. Via Lohegaon,
Pune - 412105
Maharashtra (India)

Date: / / 2025

CERTIFICATE

This is to certified that Samyak Jeetendra Shinde
A student of **BCA (AIML)** URN No 2023-B-23062005 has
Successfully Completed the Dashboard Report On

"MLBB Hero Recommender "

As per the requirement of
Ajeenkya DY Patil University, Pune was carried out under my supervision.
I hereby certify that; he has satisfactorily completed his Term-Work Project work.

Place: - Pune

Examiner

Abstract

In the realm of competitive gaming, effective hero selection can significantly influence game outcomes. This project introduces an AI-powered MLBB (Mobile Legends: Bang Bang) Hero Recommendation and Visualization System that assists players in making informed choices. By integrating backend APIs with an interactive frontend, the system delivers personalized lane-based recommendations, comparative analytics, and multiple visualizations such as pick rate trends and role distribution. It utilizes FastAPI, Streamlit, and machine learning libraries like Scikit-learn to provide a responsive, data-driven user experience. The project serves as a comprehensive example of how AI and data visualization can enhance gaming strategies and decision-making.

CHAPTER 1 – INTRODUCTION

Mobile Legends: Bang Bang (MLBB) is one of the most popular multiplayer online battle arena (MOBA) games, featuring over a hundred unique heroes. With various roles like tank, assassin, fighter, and support—choosing the most effective hero based on one's lane and team composition is crucial. Manual selection is often time-consuming and error-prone. Hence, we propose an automated recommendation and analytics tool to support informed hero selection.

Objectives:

- Build a data-driven hero recommendation engine
- Analyze hero statistics and trends
- Enable comparisons across heroes based on multiple attributes
- Provide user-friendly charts and dashboards

CHAPTER 2 – REVIEW OF LITERATURE

Past research in game analytics and recommendation systems explores both collaborative and content-based filtering models. For example, Netflix's recommendation engine merges both to enhance accuracy. In the gaming context, content-based filtering has proven effective when user histories are unavailable. Studies by Lops et al. (2011) and others emphasize the role of metadata in providing relevant suggestions. Our project adopts content-based filtering using hero statistics like win rate, pick rate, offense, defense, and skill effects. The frontend employs visual analytics to bridge the gap between data and decision-making.

CHAPTER 3 – METHODOLOGY

Dataset: mlbb_heros.csv with over 20 columns related to each hero’s attributes, such as role, lane, physical and magical stats, difficulty, win rate, and pick rate. Source: https://www.kaggle.com/datasets/sadkoktaybicici/mobile-legends-bang-bang-mlbb-heros-dataset

Tools Used:

- Python 3.11
- Streamlit
- Scikit-learn: KNN model for similarity-based recommendations
- Pandas & NumPy: Data loading and preprocessing
- Matplotlib & Seaborn: Data visualization

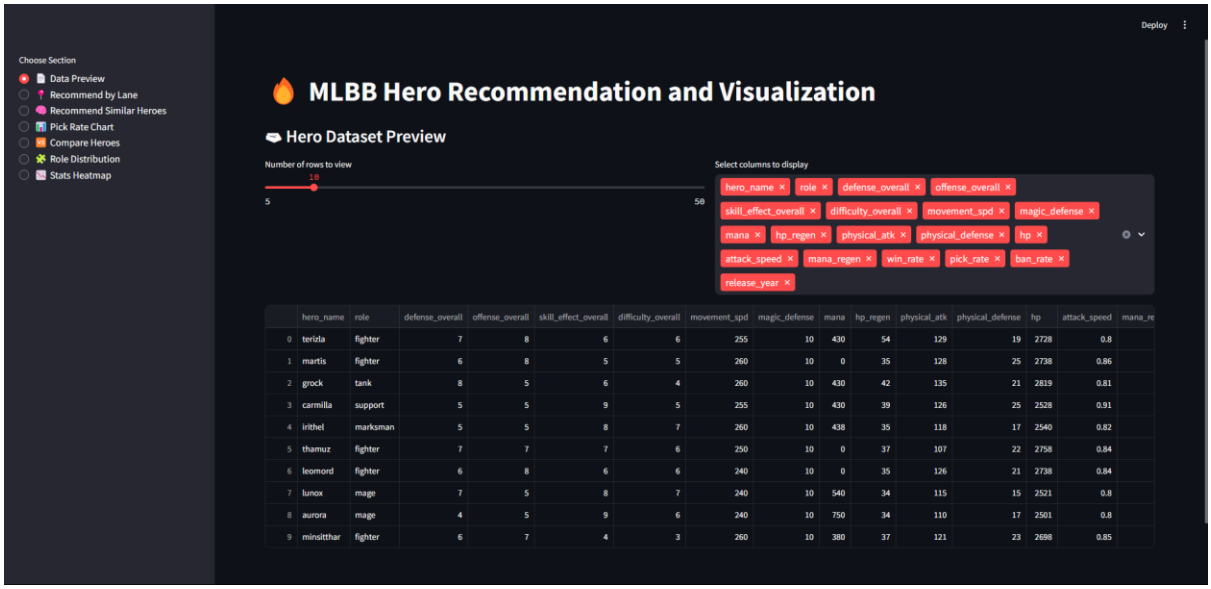
Steps:

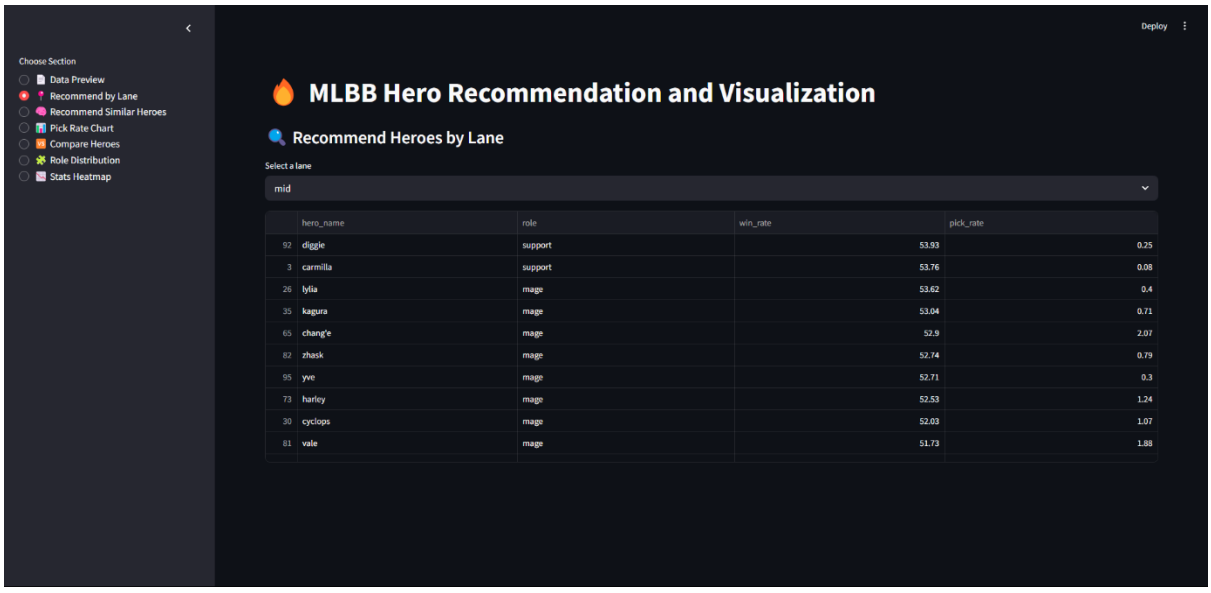
1. Data Preprocessing: Clean data, normalize text inputs (hero names, roles, lanes)
2. API Creation: Design endpoints for recommendations, comparisons, and chart rendering

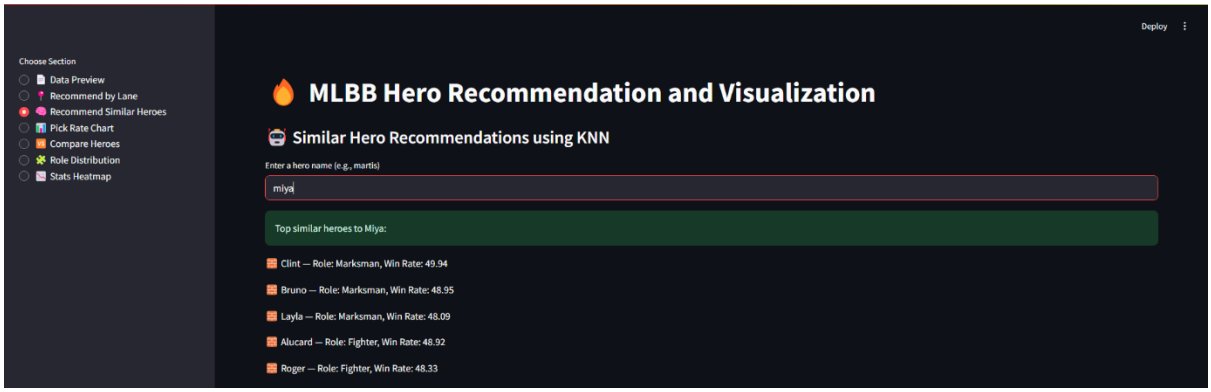
3. Visualization: Use Seaborn/Matplotlib to create pick rate charts, role distributions, and heatmaps
4. Frontend: Integrate all functionality using Streamlit widgets and layout controls

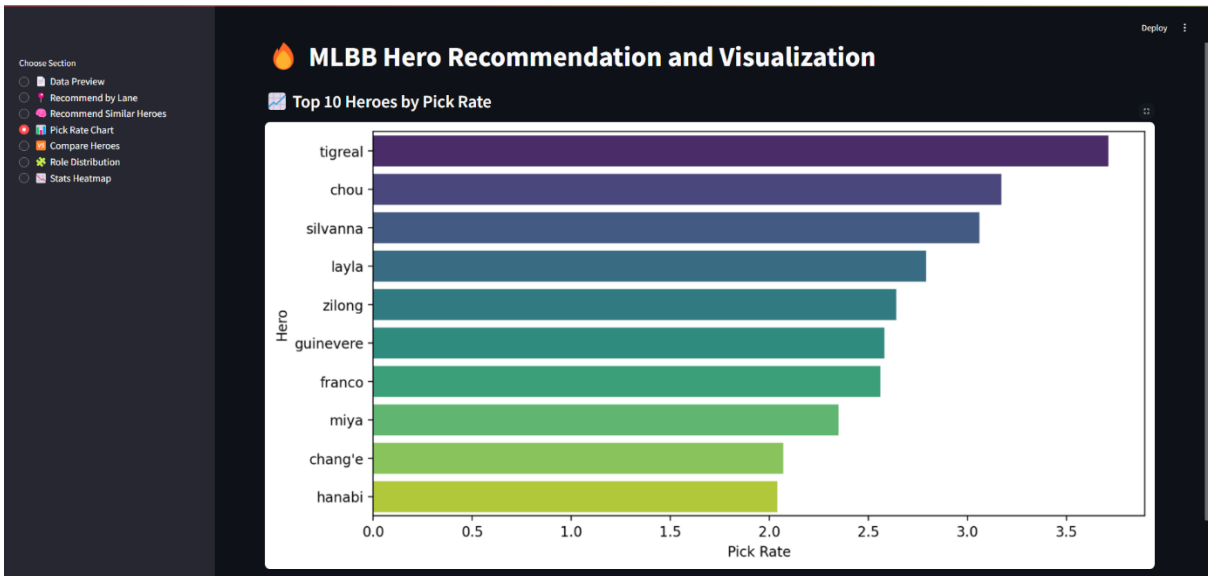
CHAPTER 4 – Implementation

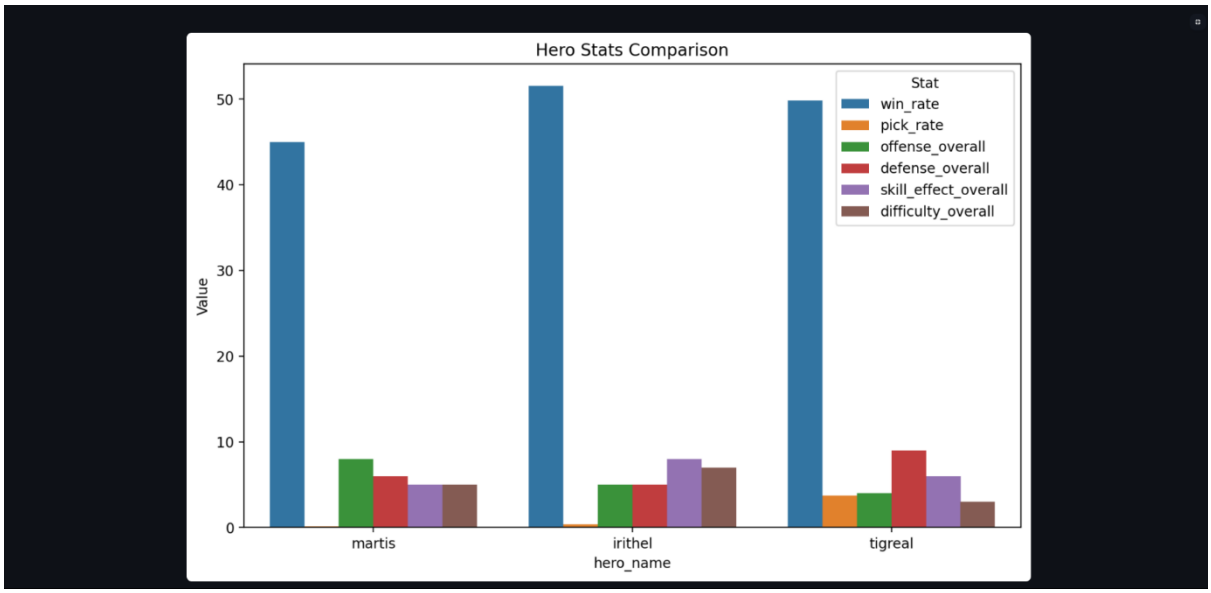
App while Running on Web:

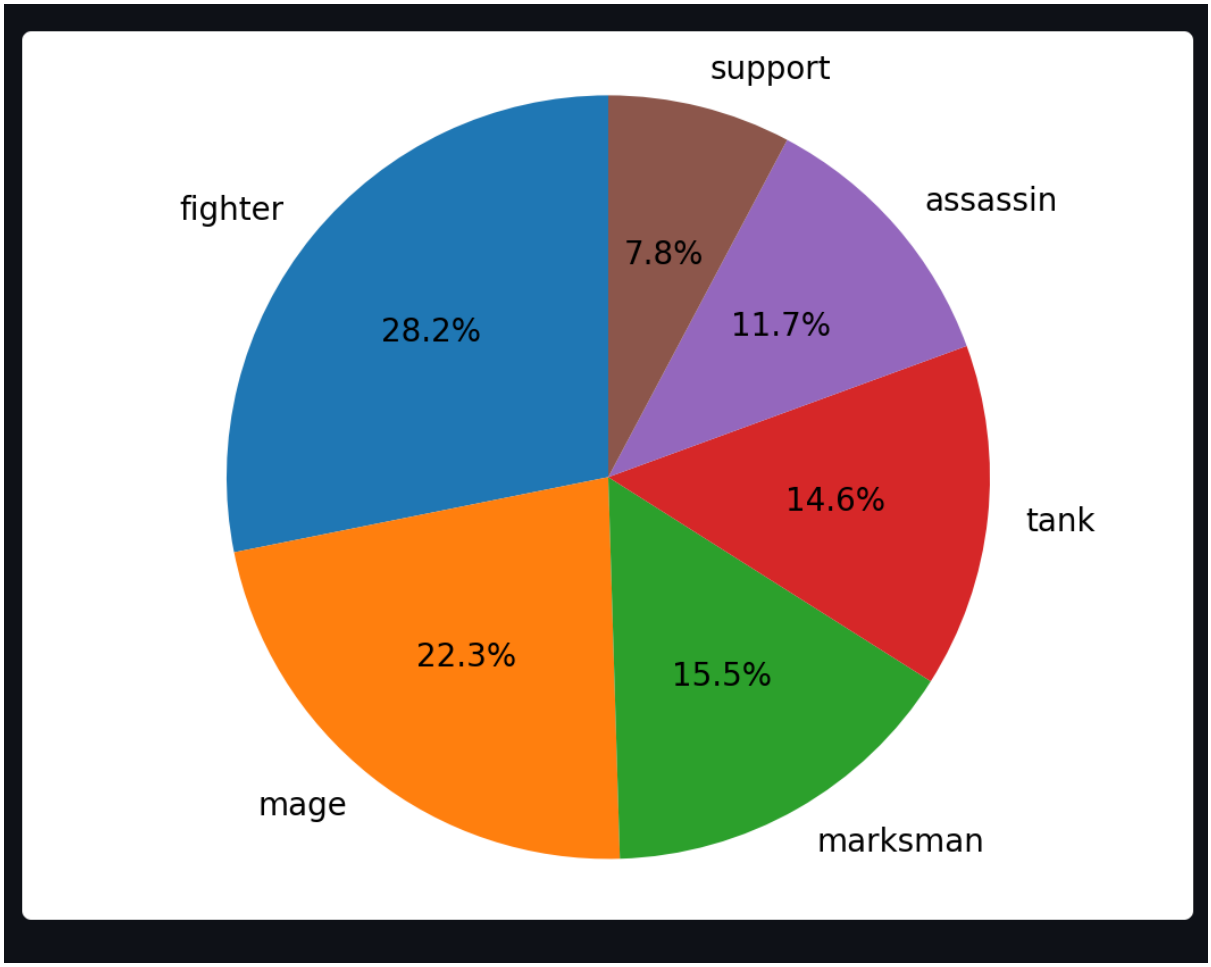


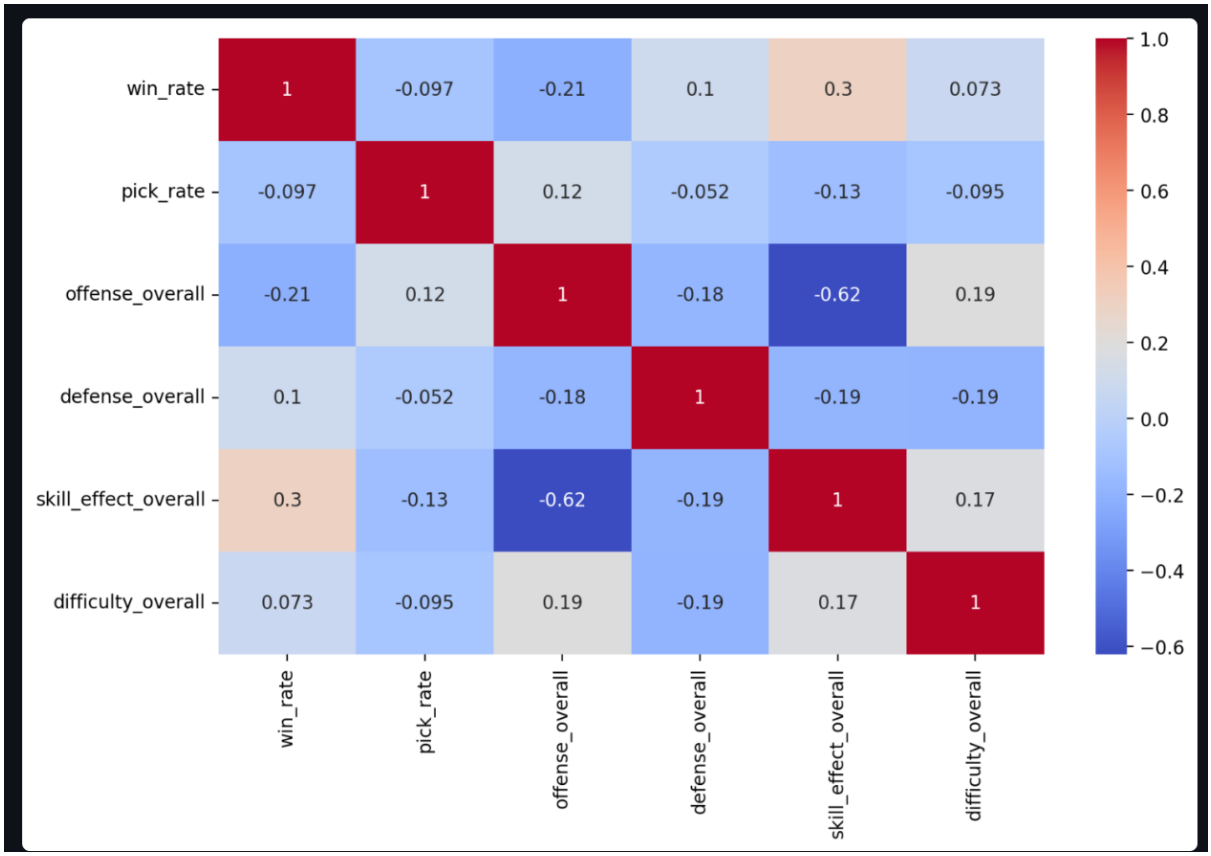



Deploy
Choose Section
Data Preview
Recommend by Lane
Recommend Similar Heroes
Pick Rate Chart
Compare Heroes
Role Distribution
Stats Heatmap
MLBB Hero Recommendation and Visualization
Similar Hero Recommendations using KNN
Enter a hero name (e.g., martis)
miya
Top similar heroes to Miya:
Clint — Role: Marksman, Win Rate: 49.94
Bruno — Role: Marksman, Win Rate: 48.95
Layla — Role: Marksman, Win Rate: 48.09
Alucard — Role: Fighter, Win Rate: 46.92
Roger — Role: Fighter, Win Rate: 48.33


Deploy
Choose Section
Data Preview
Recommend by Lane
Recommend Similar Heroes
Pick Rate Chart
Compare Heroes
Role Distribution
Stats Heatmap
MLBB Hero Recommendation and Visualization
Top 10 Heroes by Pick Rate
tigreal
chou
silvanna
layla
zilong
guinevere
franco
miya
chang'e
hanabi
Hero
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Pick Rate


Hero Stats Comparison
Stat
win_rate
pick_rate
offense_overall
defense_overall
skill_effect_overall
difficulty_overall
Value
0
10
20
30
40
50
martis
irithel
tigreal
hero_name


support
assassin
fighter
7.8%
11.7%
28.2%
14.6%
tank
22.3%
15.5%
mage
marksman


win_rate
pick_rate
offense_overall
defense_overall
skill_effect_overall
difficulty_overall
1 -0.097 -0.21 0.1 0.3 0.073
-0.097 1 0.12 -0.052 -0.13 -0.095
-0.21 0.12 1 -0.18 -0.62 0.19
0.1 -0.052 -0.18 1 -0.19 -0.19
0.3 -0.13 -0.62 -0.19 1 0.17
0.073 -0.095 0.19 -0.19 0.17 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

The above images depict the following features respectively:

1. Dataset preview
2. Lane Based hero recommendation
3. Hero recommendation based on similarity using knn model
4. Top 10 Hero by their Pick rate
5. Distribution of roles for heros
6. Correlation of the stats

CHAPTER 5 – RESULTS, SUMMARY AND RECOMMENDATIONS

5.1 Summary

- The system processes and normalizes all hero metadata
- Recommendations and comparisons are generated dynamically
- All visualizations are served as image responses and embedded directly in the UI
- Streamlit offers a minimal yet interactive experience for users

5.2 Recommendations

- Add avatars or thumbnails for each hero
- Enable filtering by difficulty or win rate thresholds
- Integrate real-time MLBB API data for meta updates

5.3 Scope for Further Research

- Apply deep learning techniques (e.g., LSTM or Transformers) for win prediction
- Combine content-based and collaborative filtering for hybrid recommendations
- Integrate player profiles and match history for personalization

5.4 Suggestions

- Expand dataset with hero abilities, crowd control and healing stats
- Enable saving or exporting hero picks and comparisons
- Include multilingual support for regional players

CHAPTER 6 –Code And LINKS

GitHub Repository: https://github.com/Chikuu7/MLBB-RECOMMENDATION-.git

Code:

```
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.neighbors import NearestNeighbors
st.set_page_config(page_title="MLBB Hero Recommender", layout="wide")
st.title("🔥 MLBB Hero Recommendation and Visualization")
# Load data
@st.cache_data
def load_data():
    df = pd.read_csv("mlbb_heros.csv")
    df.columns = df.columns.str.strip().str.lower()
    df["hero_name"] = df["hero_name"].str.strip().str.lower()
    df["role"] = df["role"].str.strip().str.lower()
    return df

df = load_data()
@st.cache_resource
def get_knn_model(data: pd.DataFrame):
    features = data[[
```

```
        "win_rate", "pick_rate", "offense_overall",
        "defense_overall", "skill_effect_overall", "difficulty_overall"
    ]]
    model = NearestNeighbors(n_neighbors=6,
metric="euclidean")
    model.fit(features)
    return model

knn_model = get_knn_model(df)

# Sidebar Navigation
option = st.sidebar.radio("Choose Section", [
    "📄 Data Preview", "📍 Recommend by Lane", "🧠
Recommend Similar Heroes",
    "📊 Pick Rate Chart", "🆚 Compare Heroes", "🧩 Role
Distribution", "🔥 Stats Heatmap"
])

# 📄 Data Preview
if option == "📄 Data Preview":
    st.subheader("📄 Hero Dataset Preview")

    col1, col2 = st.columns(2)
    with col1:
        num_rows = st.slider("Number of rows to view", 5, 50, 10)
    with col2:
        show_cols = st.multiselect("Select columns to display",
df.columns.tolist(), default=df.columns.tolist())
```

```
    st.dataframe(df[show_cols].head(num_rows))

# 📍 Recommend by Lane
# 📍 Recommend by Lane (Role-based logic)
elif option == "📍 Recommend by Lane":
    st.subheader("🔍 Recommend Heroes by Lane")

    lane = st.selectbox("Select a lane", ["gold", "mid", "jungle",
"roam", "exp"])
    lane = lane.strip().lower()

    role_map = {
        "gold": ["marksman"],
        "mid": ["mage", "support"],
        "roam": ["tank", "support"],
        "jungle": ["assassin", "fighter"],
        "exp": ["fighter", "tank"]
    }

    if lane in role_map:
        filtered = df[df["role"].isin(role_map[lane])]
        if not filtered.empty:
            st.dataframe(filtered[["hero_name", "role", "win_rate",
"pick_rate"]].sort_values(by="win_rate", ascending=False))
        else:
            st.warning("No heroes found for that lane.")
    else:
        st.error("Invalid lane selected.")
```

```
elif option == "🧠 Recommend Similar Heroes":
   st.subheader("🤖 Similar Hero Recommendations using KNN")
   hero_input = st.text_input("Enter a hero name (e.g., martis)").strip().lower()

   if hero_input:
      if hero_input in df["hero_name"].values:
         hero_index = df[df["hero_name"] == hero_input].index[0]
         distances, indices = knn_model.kneighbors(
            df.loc[[hero_index], ["win_rate", "pick_rate", "offense_overall",
                                  "defense_overall", "skill_effect_overall", "difficulty_overall"]]
         )

         st.success(f"Top similar heroes to {hero_input.title()}:")
         for idx in indices[0][1:]:
            hero_row = df.iloc[idx]
            st.write(f"🧱 {hero_row['hero_name'].title()} — Role: {hero_row['role'].title()}, Win Rate: {hero_row['win_rate']}")
      else:
         st.error("Hero not found.")


# 📊 Pick Rate Chart
elif option == "📊 Pick Rate Chart":
   st.subheader("📈 Top 10 Heroes by Pick Rate")
```

```
    top = df.sort_values(by="pick_rate",
ascending=False).head(10)
    fig, ax = plt.subplots(figsize=(10, 5))
    sns.barplot(x=top["pick_rate"], y=top["hero_name"], ax=ax,
palette="viridis")
    ax.set_xlabel("Pick Rate")
    ax.set_ylabel("Hero")
    st.pyplot(fig)

# 🆚 Compare Heroes
elif option == "🆚 Compare Heroes":
    st.subheader("📊 Compare Hero Stats")
    hero_names = st.text_input("Enter hero names (comma
separated)", "martis, irithel, tigreal")
    names = [name.strip().lower() for name in
hero_names.split(",")]
    compare = df[df["hero_name"].isin(names)]

    if not compare.empty:
        stats = ["win_rate", "pick_rate", "offense_overall",
"defense_overall", "skill_effect_overall", "difficulty_overall"]
        melted = compare.melt(id_vars="hero_name",
value_vars=stats, var_name="Stat", value_name="Value")
        fig, ax = plt.subplots(figsize=(10, 6))
        sns.barplot(data=melted, x="hero_name", y="Value",
hue="Stat", ax=ax)
        ax.set_title("Hero Stats Comparison")
        st.pyplot(fig)
    else:
        st.warning("One or more heroes not found.")
```

```
# 🧩 Role Distribution
elif option == "🧩 Role Distribution":
    st.subheader("📊 Role Distribution")
    role_counts = df["role"].value_counts()
    fig, ax = plt.subplots()
    ax.pie(role_counts, labels=role_counts.index,
autopct="%1.1f%%", startangle=90)
    ax.axis("equal")
    st.pyplot(fig)

# 📉 Stats Heatmap
elif option == "📉 Stats Heatmap":
    st.subheader("📉 Correlation Between Stats")
    stats = df[["win_rate", "pick_rate", "offense_overall",
"defense_overall", "skill_effect_overall", "difficulty_overall"]]
    corr = stats.corr()
    fig, ax = plt.subplots(figsize=(10, 6))
    sns.heatmap(corr, annot=True, cmap="coolwarm", ax=ax)
    st.pyplot(fig)
```