```
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
import spacy
import matplotlib.pyplot as plt
import seaborn as sns

# Load the medical transcription dataset
df_medical = pd.read_csv("mtsamples.csv")

# Display basic information about the dataset
print("Dataset Structure:")
print(df_medical.info())

# Display basic statistics about the numerical columns
print("\nDataset Statistics:")
print(df_medical.describe())

# Display the unique values in the 'medical_specialty' column
print("\nMedical Specialties:")
print(df_medical['medical_specialty'].unique())

# Text cleaning
df_medical['cleaned_text'] = df_medical['transcription'].apply(lambda x: ' '.join([word.lower() for
word in str(x).split() if word.isalnum()]))

# Handling missing values and duplicates
df_medical.dropna(subset=['cleaned_text'], inplace=True)
df_medical.drop_duplicates(subset='cleaned_text', inplace=True)

# Splitting the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(df_medical['cleaned_text'],
df_medical['medical_specialty'], test_size=0.2, random_state=42)

# Feature extraction using TF-IDF
tfidf_vectorizer = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_val_tfidf = tfidf_vectorizer.transform(X_val)
```

```python
# Train/Fine-tune the model
clf = RandomForestClassifier()
clf.fit(X_train_tfidf, y_train)

# Evaluate the model
y_pred = clf.predict(X_val_tfidf)
print("Classification Report:")
print(classification_report(y_val, y_pred))

# Incorporate a language model (spaCy) for tokenization
nlp = spacy.load("en_core_web_sm")

# Tokenize text using spaCy
def tokenize_text(text):
    doc = nlp(text)
    return [token.text for token in doc]

# Apply tokenization to 'transcription' column
df_medical['tokens'] = df_medical['transcription'].apply(tokenize_text)

# Exploratory Data Analysis (EDA)
# Visualize the distribution of medical specialties
plt.figure(figsize=(12, 6))
sns.countplot(y='medical_specialty', data=df_medical)
plt.title('Distribution of Medical Specialties')
plt.xlabel('Count')
plt.show()

# Visualize the most common words in the dataset
common_words = pd.Series(' '.join(df_medical['cleaned_text']).split()).value_counts()[:10]
common_words.plot(kind='bar', figsize=(12, 6))
plt.title('Top 10 Most Common Words in Transcriptions')
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.show()

# Visualize the confusion matrix
conf_matrix = pd.crosstab(y_val, y_pred, rownames=['Actual'], colnames=['Predicted'])
plt.figure(figsize=(12, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.show()
```

Documentation

1. Dataset Understanding:
   - Loaded the medical transcription dataset ('mtsamples.csv').
   - Displayed basic information about the dataset, including features and labels.
   - Displayed basic statistics about the numerical columns.
   - Displayed unique values in the 'medical_specialty' column.

2. Data Preprocessing:
   - Cleaned text by removing special characters, lowercasing, and tokenization.
   - Handled missing values and duplicates.
   - Split the data into training and validation sets.

3. Train/Fine-tune on given domain-specific dataset:
   - Extracted features using TF-IDF.
   - Trained a RandomForestClassifier.

4. Incorporate a language model:
   - Incorporated spaCy for tokenization.

5. Evaluate the effectiveness:
   - Evaluated the model using the classification report.

6. EDA on the train data and test results:
   - Explored the distribution of medical specialties.
   - Visualized the most common words in transcriptions.
   - Visualized the confusion matrix.

7. Challenges and Solutions:
   - Handled missing values and duplicates to ensure clean data.
   - Adapted the code to handle variations in the dataset structure.
   - have trouble with the spacy library.

8. Results:
   - Achieved insights into the distribution of medical specialties and common words.
   - Evaluated the model's performance using the confusion matrix.

9. Future Considerations:
   - Further fine-tuning of the model for improved performance.
   - Exploration of advanced language models for better feature extraction.