**Milestone 1: Project Inception, Model Card for my personal AI in Digital Transformation and Machine Learning Project**

**Overview and Business Case Description:**

This milestone outlines the initial phase in developing a machine learning system to analyze sentiments related to the US Elections 2024, transitioning from the 2020 elections analysis model.

This model card represents a comprehensive guide for an AI model designed for digital transformation and machine learning projects, focusing on sentiment analysis of tweets related to the US Elections 2024.

The model is specifically tailored for application in the United States and not suitable for regions like Morocco due to differences in linguistic, cultural, constitutional and political contexts.

The transition from analyzing the 2020 US Elections to the 2024 Elections using machine learning involves several key steps and considerations to ensure the relevance and effectiveness of the AI model. This model card outlines a machine learning system for analyzing sentiment related to the US Elections 2024, similar to what was done for the 2020 elections. Both utilize the BERT language model to understand sentiments from Twitter data, which indicates a methodological alignment.

BERT (Bidirectional Encoder Representations from Transformers) is a groundbreaking model in the field of Natural Language Processing (NLP) developed by Google. The innovative aspect of BERT is its deep bidirectionality, which allows the model to understand the context of a word based on all of its surroundings (left and right of the word).

**Proposed Pipeline that includes the DataScientist work:**

1. Data Collection, Pre-processing and Update
2. Model Re-training and Fine-tuning
3. Sentiment Analysis Adaptation
4. Incorporating New Political Context
5. Feature Extraction and contextual understanding
6. Enhanced Performance Metrics
7. Stakeholder Feedback Integration
8. Testing and Validation
9. Deployment and Monitoring and Maintenance
10. **Project Scope and Objective, Project Name:**

* AI-2024-US-Election-Sentiment-Analyzer Goal: Adapt and enhance the existing 2020 election sentiment analysis model for the 2024 US Elections.
* Model Description Model Name: AI-2024-US-Election-Sentiment-Analyzer
* Model Type: Sentiment Analysis using pre-trained BERT.
* Primary Use: Analysis of public sentiment in tweets related to US Elections 2024
* Language: English
* Region of Application: United States (Not applicable for Morocco)
* Architecture: Based on BERT (Bidirectional Encoder Representations from Transformers)
* Training Dataset: Over 10 million English tweets regarding US political discussions, specifically filtered for US Elections 2024 topics.
* Target Audience: Political analysts, campaign strategists, digital marketers, and researchers
* Use Cases: Predicting election outcomes, analyzing public opinion trends, enhancing targeted political campaigning.

1. **Feasibility, Data Requirements and Specificities:**

The model is based on advanced NLP techniques and leverages pre-trained models like BERT for effective sentiment analysis from Tweets.

Its feasibility is grounded in current technological capabilities and available computational resources.

* Data Source: Twitter, focusing on tweets related to US Elections 2024.
* Data Volume: A substantial dataset of over 10 million tweets.
* Data Diversity: Inclusion of tweets across various demographics, regions, and political affiliations in the US.
* Data Privacy: Adherence to data privacy laws and ethical guidelines, especially regarding user information.

**Actual model details (US Elections 2020):**

* **Model Name: Pre-trained BERT on Twitter US Political Election 2020.**
* **Primary Use: Stance Detection, with the ability to be fine-tuned for various downstream tasks such as classification.**
* **Pre-training Base: BERT-base (uncased), known as bert-base-uncased.**
* **Data Source: The model was pre-trained on a corpus of over 5 million English tweets related to the 2020 US Presidential Election.**
* **Purpose: Capture the linguistic and contextual nuances of political discourse on Twitter during the election period.**
* **Initial Training: The model was initialized with BERT-base weights and trained with a standard Masked Language Model (MLM) objective. This involves predicting masked tokens in a sentence, which helps the model learn context.**
* **Usage and Functionality**
* **Fine-tuning Capability: The pre-trained model can be fine-tuned for specific tasks like sentiment analysis or classification based on the user’s dataset and needs.**
* **Masked Language Model: It can be used directly with the fill-mask pipeline to predict the masked words in a given sentence, thereby understanding context and language usage in political discussions.**
* **Implementation Details Based on the Repository**
* **GPU Utilization: The model leverages GPU acceleration if available for faster computation.**
* **Model Path: The path kornosk/bert-political-election2020-twitter-mlm refers to the model's location on Hugging Face, a platform hosting the model weights and tokenizer for public access.**
* **Tokenization and Model Loading: The code uses BertTokenizer for converting text into tokens that BERT understands, and BertForMaskedLM to load the pre-trained Masked Language Model.**
* **Fill Mask Pipeline: An NLP pipeline for filling in the masked word in a sentence is created using Hugging Face's pipeline function with the fill-mask task.**
* **Model Output: For a given sentence with a [MASK] token, the model outputs a list of possible tokens that can fill the mask, along with confidence scores.**
* **Sample Code Execution:**

**Example Usage: The code snippet demonstrates filling in the masked word in a sample sentence related to Trump. The outputs are the model's predictions for the masked token.**

* **Embedding Extraction: Optionally, the code can be used to extract the model's internal embeddings for a given input, which can be beneficial for various NLP tasks.**

**Actual Dataset for Research Purposes:**

* Data Sets and Link to the dataset model and its results:

<https://huggingface.co/kornosk/bert-political-election2020-twitter-mlm/tree/main>

<https://github.com/GU-DataLab/stance-detection-KE-MLM>

<https://huggingface.co/kornosk/bert-election2020-twitter-stance-biden>

<https://www.researchgate.net/publication/354427394_Biden_vs_Trump_Modelling_US_general_elections_using_BERT_language_model>

<https://www.researchgate.net/publication/373453545_Sentiment_Analysis_of_Tweets_Before_the_2024_Elections_in_Indonesia_Using_BERT_Language_Models> (Indonesian case)

* Format: CSV with three columns: "tweet\_id", "text", "label"

Labels: {0: "AGAINST", 1: "FAVOR", 2: "NONE"}

Size and Content: The dataset contains 2500 manually-stance-labeled tweets, 1250 for each candidate (Joe Biden and Donald Trump).

These were sampled from a larger set of over 5 million English tweets related to the 2020 US Presidential Election.

* Collection Period: January 2020 to September 2020
* Data Source: Twitter Streaming API, focusing on election-related hashtags and keywords.
* Labeling Process: Amazon Mechanical Turk

Available Models:

* BERT-Political-Election-2020-Twitter-MLM
* BERT-Election-2020-Twitter-Stance-Biden-f-BERT
* BERT-Election-2020-Twitter-Stance-Biden-KE-MLM
* BERT-Election-2020-Twitter-Stance-Trump-f-BERT
* BERT-Election-2020-Twitter-Stance-Trump-KE-MLM
* Model Hosting: All models are hosted on Huggingface for easy access.

(Within the previous links, the mentioned by the author model card presents a BERT-based language model that has been specifically pre-trained and fine-tuned for stance detection in relation to Joe Biden, using a dataset of over 5 million English tweets from the 2020 US Presidential Election. This model, referred to as f-BERT, is initialized with the BERT-base architecture and further trained to identify whether a tweet expresses a stance that is against, in favor of, or neutral towards Joe Biden.

The f-BERT model is designed to process tweets, evaluate their content, and categorize the stance of the tweet with respect to Biden. The model utilizes the AutoTokenizer and AutoModelForSequenceClassification classes from the transformers package, and is intended to run on a GPU for optimal performance.

Three example usages are provided, demonstrating how to predict the stance of sample sentences using the model. The predictions are made by tokenizing the input sentence, passing it through the model, and then interpreting the softmax probabilities of the output to assign a stance label.

The authors encourage citation of their paper, which details the Knowledge Enhanced Masked Language Model for Stance Detection, presented at the NAACL 2021 conference, if the model or its concepts are used beneficially in other works.)

Usage and Compatibility:

* Tested Environments: Pytorch v1.8.1 and Transformers v4.5.1
* Model Usage: Detailed instructions available on specific model pages.
* Sample Use Case:

Loading and prediction example provided in the documentation.

1. **Justification of the model and performance metrics:**

* The model aims to analyze public sentiment towards political candidates or issues, providing valuable insights for stakeholders in political campaigns, media, and academic research.
* Baseline Model Performance: Establishing a baseline using a pre-existing sentiment analysis model, targeting an initial accuracy of 85%.
* Industry Standard Comparison: Comparing the model’s performance with existing sentiment analysis tools used in political campaigning.
* Historical Data Performance: Evaluating the model against past election data to assess predictive reliability.

**Performance Metrics:**

* Baseline Accuracy: 85% sentiment classification accuracy
* Business Impact Metrics: Improved decision-making in political investments, measurable ROI increase in campaign strategies.
* Update Frequency: Quarterly retraining with updated datasets
* Monitoring Metrics: Accuracy, precision, recall, and real-time performance in sentiment classification

1. **Applicability:**

This model is tailored for the US context, utilizing a dataset comprising English tweets. Cultural and linguistic nuances specific to the US make it less applicable to regions with different languages and political landscapes, like Morocco.

The model performs sentiment analysis on tweets, classifying them into categories like positive, negative, or neutral. This provides an understanding of public opinion trends.

Training Data

Initial Training:

The model was initially pre-trained on a dataset comprising over 5 million English tweets from the 2020 US Presidential Election.

Retraining and Fine-tuning:

For the 2024 elections, the model has been further fine-tuned using a new dataset, including sentiments and stances relevant to new candidates and political issues.

Training Objective

* Base Model: Initialized with the BERT-base architecture.
* Objective: Enhanced for stance detection with a focus on the dynamics of the 2024 US Elections, replacing the initial objective centered around Joe Biden.

Usage and Implementation:

* Specific Adaptation: This model is now fine-tuned for a broader range of political figures and issues pertinent to the 2024 elections.
* Technical Integration:
* Utilize AutoTokenizer and AutoModelForSequenceClassification from the transformers package.
* Leverage GPU capabilities for enhanced performance.
* The model can be loaded from "kornosk/bert-election2024-twitter-stance" (hypothetical path).

Sample Predictions

* Label Mapping: {0: "AGAINST", 1: "FAVOR", 2: "NONE"}
* Usage Example: Demonstrated through Python scripts, predicting sentiment stances like "AGAINST", "FAVOR", or "NEUTRAL" for input sentences.

Model Evolution for 2024

* Transitioning Focus: From the initial stance detection towards specific candidates like Joe Biden in 2020, to a more expansive approach capturing the diverse political landscape of the 2024 US Elections.
* Continuous Learning: The model will continue to evolve, adapting to real-time political discourse and sentiment shifts throughout the election cycle.

1. **Pipeline for Dataset Adjustment**

The model includes a pipeline for refining the dataset, involving data cleaning, normalization, and augmentation to enhance accuracy and relevance.

* Data Pipeline: Includes data cleaning, normalization, and augmentation processes.
* Retraining Process: Automated retraining pipeline, incorporating new data and adjusting parameters as needed

1. **Goals and Pipeline Building:**

To build this pipeline, goals include accurate sentiment classification, scalability, and adaptability to real-time data.

The process involves selecting appropriate NLP tools, training the model with a representative dataset, and validating its accuracy.

1. **Monitoring:**

Continuous monitoring is essential for maintaining model accuracy.

This involves regular updates to the dataset, algorithm adjustments, and performance tracking against key metrics.

1. **Model as a Component:**

The model is one of several components in a broader digital strategy, integrating with other analytics tools and data sources for comprehensive insights.

1. **Business Value:**

This AI model is positioned as a cutting-edge tool for political sentiment analysis, offering precise, real-time insights for strategic decision-making.

* Strategic Decision-Making: Provides political strategists and campaigners with insights into public opinion trends, enabling data-driven decision-making.
* Targeted Campaigning: Helps in tailoring political messages and advertisements based on public sentiment.
* Predictive Analysis: Forecasts election outcomes, aiding in investment decisions related to campaign financing and media planning.

1. **Baseline Details:**

The baseline model is established using a pre-trained BERT model, with performance metrics determined based on accuracy, recall, and precision in sentiment classification.

1. **Business Metrics:**

The following models should aim to improve accuracy, precision, and recall. Business metrics also include the impact on decision-making efficacy and ROI in political campaigns.

**Business Impact and ROI:**

* Goal: Enhancing the accuracy of election outcome predictions
* ROI Measurement: Tracking the correlation between model predictions and actual election results, and the impact on campaign ROI.
* Risk Mitigation: Reducing investment risks and maximizing returns through accurate sentiment analysis.
* Accuracy in Sentiment Classification: The primary metric; the model's effectiveness in accurately classifying sentiments in tweets.
* Predictive Accuracy: The model's ability to predict election outcomes based on sentiment analysis.
* ROI Impact: Measuring the impact on campaign strategies, including increased engagement, voter turnout, and effective media spending.
* User Adoption Rate: The rate at which political analysts and campaign strategists adopt and integrate the tool into their workflows.

1. **Benefit and Assessment:**

The primary benefit is the improved accuracy in predicting election outcomes, aiding in strategic planning for advertisements and campaigns.

Assessment involves comparing model predictions with actual election results and ROI metrics.

1. **Business Goals and ML:**

The goal is to leverage ML for predicting election outcomes, enabling businesses to make informed investment decisions in advertising and campaign strategies.

* **Maximizing ROI:**

By improving the accuracy of sentiment analysis, the model aids in minimizing losses and risks associated with political investments, optimizing tax strategies, and maximizing ROI.

* **Mapping Accuracy to Investment:**

The model's accuracy in sentiment analysis directly correlates with the effectiveness of investment decisions in political campaigns.

**Pipeline for Retraining and Deployment**

The model includes a pipeline for regular retraining with updated data, ensuring its relevance and accuracy.

Deployment strategies involve integration with existing digital platforms and tools.

1. **Link to MLOps:**

The model's development and maintenance adhere to MLOps best practices, ensuring efficient deployment, monitoring, and continuous improvement.

Deployment: Integrated with existing digital platforms using MLOps practices

Continuous Improvement: Regular updates and optimizations based on performance metrics and user feedback.

1. **Set of Questions for Guests:**

Questions for potential users or stakeholders might include their specific needs in sentiment analysis, expectations from the model in terms of accuracy, and desired integration capabilities with existing tools.

1. **Limitations:**

* Cultural and Linguistic Limitations: Not suitable for non-English or culturally diverse regions like Morocco
* Bias Mitigation: Continuous monitoring for biases in sentiment classification and periodic model retraining

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

from transformers import AutoTokenizer, AutoModelForSequenceClassification

import torch

import numpy as np

# Choose the most powerful available device - GPU (CUDA) if available, otherwise CPU

device = torch.device("cuda" if torch.cuda.is\_available() else "cpu")

# Define the path for the pre-trained model

# This model is specifically trained for sentiment analysis related to US Elections

pretrained\_LM\_path = "kornosk/bert-election2020-twitter-stance-biden"

# Load the tokenizer and model from the pre-trained path

tokenizer = AutoTokenizer.from\_pretrained(pretrained\_LM\_path)

model = AutoModelForSequenceClassification.from\_pretrained(pretrained\_LM\_path)

# Move the model to the chosen device

model.to(device)

# Define a dictionary to interpret model's output labels

id2label = {

0: "AGAINST",

1: "FAVOR",

2: "NONE"

}

def predict\_sentiment(sentence):

"""

Function to predict sentiment of a given sentence.

Arguments:

sentence (str): A text snippet typically a tweet or sentence.

Returns:

str: Predicted sentiment label for the given sentence.

"""

# Tokenize the sentence and convert to tensor

inputs = tokenizer(sentence.lower(), return\_tensors="pt").to(device)

# Predict sentiment with the model

with torch.no\_grad():

outputs = model(\*\*inputs)

# Calculate probabilities from model outputs and pick the most probable label

predicted\_probability = torch.softmax(outputs.logits, dim=1)[0].cpu().numpy()

predicted\_label = id2label[np.argmax(predicted\_probability)]

return predicted\_label, predicted\_probability

# Example usage

sentences = [

"Hello World.",

"Go Go Biden!!!",

"Biden is the worst."

]

# Predict sentiment for each example sentence

for sentence in sentences:

predicted\_label, probabilities = predict\_sentiment(sentence)

print(f"Sentence: '{sentence}'\nPrediction: {predicted\_label}")

print(f"Probabilities -> Against: {probabilities[0]:.4f}, Favor: {probabilities[1]:.4f}, Neutral: {probabilities[2]:.4f}\n")

# Note: Please consider citing the associated paper if you find this model useful for your research or application.