**Project Title:**

Do Job Adverts Predict Future Employment?

A Lag Analysis Using ONS Job Advert Data and Employment Rates in the East Midlands

**Objective**

This project investigates whether monthly job advert trends (a proxy for labour demand) can help predict future changes in employment rates (a measure of labour market outcomes) in the East Midlands. By aligning lagged job advert data with subsequently published employment rates, the project tests whether job adverts can serve as a leading indicator of employment outcomes. The aim is to identify whether trends in recruitment activity are predictive of real changes in employment in a specific region, using only open official data and transparent, interpretable statistical methods.

**Key Research Question**

Can increases in monthly job adverts in the East Midlands region predict corresponding increases in the employment rate one or more months later?

**Datasets Required**

**1. ONS Real-Time Indicators: Job Advert Estimates**

* Dataset: Online Job Advert Estimates (experimental RTI dataset)
* Frequency: Weekly, but also available in monthly aggregates
* Geographic Breakdown: UK, and possibly NUTS1 regions including East Midlands
* Format: CSV
* Source: https://www.ons.gov.uk/economy/economicoutputandproductivity/output/datasets/onlinejobadvertestimates

**2. ONS Labour Market Statistics: Regional Employment Rate**

* Dataset: Labour Market Profile - East Midlands or Labour Force Survey time series
* Metric: Employment Rate (aged 16 to 64)
* Frequency: Monthly or quarterly (most detailed local data is quarterly)
* Geography: East Midlands (NUTS1), Lincolnshire if available
* Format: CSV
* Source: https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/employmentandemployeetypes/datasets/labourmarketstatistics

**Methodology**

**Step 1: Data Preparation**

* Load the monthly job adverts and employment rate datasets into Python using pandas.
* Clean the datasets: parse date columns, handle missing values, ensure values are numeric.
* Convert job adverts from weekly to monthly aggregates if necessary (e.g., by summing or averaging over calendar months).
* If employment data is quarterly, interpolate to monthly estimates (e.g., linear interpolation or hold forward previous known value).
* Align time indices and truncate both series to cover a shared time window.

**Step 2: Lag Construction**

* Create lagged job advert series:
  + job\_adverts[t-1]: previous month’s adverts
  + job\_adverts[t-2]: two months prior
  + job\_adverts[t-3]: three months prior
* These will serve as independent variables when predicting employment rate at time t.

**Step 3: Correlation Analysis**

* Calculate Pearson correlation coefficients between:
  + employment\_rate[t] and job\_adverts[t]
  + employment\_rate[t] and job\_adverts[t-1]
  + employment\_rate[t] and job\_adverts[t-2]
  + employment\_rate[t] and job\_adverts[t-3]
* This will show whether job adverts correlate more strongly with future employment levels, supporting the hypothesis that they act as a leading indicator.

**Step 4: Simple Linear Regression**

* Run four separate ordinary least squares regressions of the form:
  + employment\_rate[t] = α + β \* job\_adverts[t-k] + ε
  + For k in [0, 1, 2, 3]
* Extract coefficients, p-values, and R² from the regression summary.
* Assess whether lagged values of job adverts explain variance in the employment rate.

**Step 5: Visualization**

* Line plots showing:
  + Monthly job adverts and employment rate over time (on dual axes)
  + Lagged adverts and employment trends for visual comparison
* Scatter plots of employment\_rate[t] against each job\_adverts[t-k]
* Correlation bar chart: correlation values at lags 0 to 3

**Interpretation Framework**

* A significant, positive correlation at lag-1 or lag-2 would suggest that job adverts can be used to anticipate changes in employment.
* A lack of correlation or inconsistent results across lags might suggest the relationship is coincidental or masked by external factors.
* Regression results should include confidence intervals and p-values to assess the strength and significance of the relationship.
* Any strong result should be discussed critically, acknowledging data lag, seasonal variation, and limitations in causal inference.

**Deliverables**

1. Clean, aligned time-series dataset combining job adverts and employment rate.
2. Code scripts in Jupyter Notebook showing data loading, lag construction, correlation and regression.
3. Plots of employment vs job adverts with and without lags.
4. Analytical summary of key findings.
5. Optional Streamlit dashboard to interactively explore the data and test different lags.
6. Written report or blog post summarising methodology and implications (for a general audience).

**Significance**

This project demonstrates a practical use of real-time and delayed economic indicators to test a causal hypothesis about labour market dynamics. It reflects awareness of official statistics, sound statistical reasoning, and an ability to work with real-world messy datasets. It is directly relevant to regional planning, economic monitoring, and workforce analysis, and it could easily be extended to other regions or datasets (e.g., comparing across regions or adding vacancy/claimant count).

**Tools and Technologies**

* Python
* pandas for data manipulation
* matplotlib/plotly for visualisation
* statsmodels for regression
* seaborn for statistical plots (optional)
* Streamlit (optional, for dashboard)

**Timeline (2–3 Weeks)**

**Week 1:**

* Collect and clean datasets
* Align job adverts and employment data
* Construct lagged features

**Week 2:**

* Perform correlation and regression analysis
* Visualise relationships
* Interpret results and prepare a written summary

**Optional Week 3:**

* Build Streamlit dashboard
* Write formal blog post or portfolio summary

This project balances technical skill with local and public policy relevance. It is well-suited for academic portfolios, university applications, or interviews that assess initiative, analytical depth, and understanding of socio-economic dynamics.

**ONS EconBot: AI-Powered Assistant for Exploring the UK Economy in Real Time**

**1. Overview**

**ONS EconBot** is an interactive AI assistant built using publicly available data from the UK’s Office for National Statistics (ONS). It enables users to ask natural language questions about the UK economy and receive accurate, relevant, and clearly written responses drawn from ONS’s *Real-Time Indicators of Economic Activity* bulletins.

The project automates the scraping, processing, semantic indexing, and summarization of these weekly and monthly bulletins. It combines modern natural language processing techniques, including sentence embeddings, semantic similarity search, large language models (LLMs), and sentiment analysis.

This project was designed and built as a summer research initiative to showcase how machine learning can be applied to improve access to government data and public economic information.

**2. Problem Statement**

The ONS publishes regular bulletins that contain a wide range of high-value economic indicators: job adverts, card spending, gas consumption, debit failure rates, retail footfall, and more. While these bulletins are informative and well-researched, they suffer from the following problems:

* **Difficult to access and interpret:** Each bulletin can be several thousand words long, requiring careful reading to extract key points.
* **No search or query interface:** Users cannot ask targeted questions in plain English.
* **No summarization layer:** Users looking for quick, high-level summaries must read entire sections manually.
* **No real-time trend insight:** There is no automated way to track sentiment or tone shifts across weeks.

This project addresses those challenges by creating an interface where users can engage with these bulletins conversationally, with immediate and reliable answers generated from the official data.

**3. Project Goals**

* To make real-time ONS economic bulletins **searchable by natural language**.
* To enable **automated summarization** of economic data using generative language models.
* To track and summarize **economic sentiment** across multiple bulletins.
* To demonstrate how **open data** can be combined with **open-source NLP tools** to create a civic-facing AI application.

**4. How It Works**

**Step 1: Data Collection**

* The assistant scrapes historical bulletin pages from the ONS archive (over 200 bulletins).
* It uses requests and BeautifulSoup to extract structured text, bullet points, and metadata.
* All content is cleaned and saved in JSONL format in Google Drive for reuse.

**Step 2: Paragraph Segmentation**

* Each bulletin is split into smaller paragraphs using regular expressions based on double newlines and section headers.
* Paragraphs are filtered to ensure a minimum length for semantic analysis.

**Step 3: Semantic Embedding**

* Each paragraph is converted into a dense vector using sentence-transformers with the model all-MiniLM-L6-v2.
* This allows for **meaning-based matching** rather than keyword matching.

**Step 4: FAISS Indexing**

* A FAISS vector index is built for all paragraph embeddings.
* When a user asks a question, the query is also embedded and compared to the index to retrieve the most relevant paragraphs.

**Step 5: Sentiment Analysis**

* Each paragraph is scored using the VADER sentiment analyzer.
* A compound score (-1 to +1) is assigned to capture tone.
* These scores are used to compute a rolling average and track the overall mood of recent bulletins.

**Step 6: Language Model Rewriting**

* The top-matching paragraph is passed through a lightweight language model (flan-t5-base).
* The model rewrites the paragraph to mimic the tone and style of official ONS summaries.

**Step 7: Interactive Assistant**

* The chatbot presents the user with:
  + Retrieved paragraphs with source URLs
  + A rewritten summary
  + Optionally, the overall sentiment trend

**5. Key Features**

**a. Natural Language Interface**

* Users can type plain-English queries like:
  + “How are job adverts trending?”
  + “What’s happening with consumer spending?”
  + “Is economic sentiment improving?”

**b. Rephrased Bulletins**

* Raw paragraphs are reworded using flan-t5-base to provide clean, digestible, and formal summaries.

**c. Sentiment Summarization**

* If the user includes the word “sentiment” in their question, the assistant returns a high-level mood classification:
  + Broadly positive
  + Somewhat negative
  + Mixed or neutral

**d. Citation and Traceability**

* All retrieved information includes a source URL back to the original ONS bulletin for transparency and further reading.

**6. Technologies Used**

| **Component** | **Technology** |
| --- | --- |
| Web Scraping | requests, BeautifulSoup |
| Data Storage | JSONL files on Google Drive |
| Semantic Embeddings | sentence-transformers (MiniLM) |
| Vector Search | faiss-cpu |
| Language Generation | transformers (flan-t5-base) |
| Sentiment Analysis | vaderSentiment |
| Runtime Environment | Google Colab |

**7. Example Interaction**

**User query:**

What’s going on with gas prices?

**EconBot:**

* Returns 2–3 paragraphs about recent changes in gas prices.
* Rephrases one into a clean ONS-style summary.
* Shows the source link.

**User query:**

What is the sentiment of the economy?

**EconBot:**

Sentiment over recent bulletins is somewhat negative.

**8. Impact**

This project demonstrates:

* A working, end-to-end example of applying NLP to real government datasets.
* A way to democratize access to economic information.
* An assistive tool for education, journalism, and policy review.

It shows that a student can independently:

* Scrape and structure real data
* Apply sentence embeddings and vector search
* Use LLMs for controlled summarization
* Track and report sentiment trends

**9. Limitations and Future Work**

* Currently limited to paragraph-level summaries; future versions could combine multiple paragraphs.
* Sentiment analysis uses simple rolling average; future versions may use topic-specific sentiment.
* Does not support charts or numerical data parsing yet.
* Currently relies on flan-t5-base; can be improved with fine-tuned models.

**10. Conclusion**

**ONS EconBot** is a lightweight, intelligent assistant that brings semantic search and natural language summarization to official UK economic bulletins. It is a strong demonstration of how students can use open-source tools and public data to create something impactful, usable, and technically advanced in a short period of time.

This project exemplifies applied AI for public understanding and civic benefit.

## **My Journey into Data and the ONS EconBot Project**

I’ve always been curious about how data shapes the world around us — especially when it comes to the economy and job market. This interest really started to grow as I was finishing my GCSEs. I was thinking about what kind of work I might want to do, and began searching through job adverts to see what opportunities were out there for someone like me.

As I looked through these listings each week, I started noticing patterns: some sectors had more openings than others, and some roles came up again and again. I began wondering if there was more behind these trends — questions like: *Are job adverts a sign of economic growth? Do more job postings mean more people are getting employed? Or is it the other way around?* That curiosity led me to the ONS (Office for National Statistics) website, where I discovered the **Online Job Advert Estimates** published as part of their **Real-Time Indicators** series.

This sparked the idea for my project: I wanted to explore whether job advert volumes could predict changes in the employment rate — especially in my local area, the East Midlands. It felt like a real question that mattered, and something I hadn’t seen answered before.

While working on this, I contacted the ONS a few times to better understand the datasets. During those conversations, I found out about a short work experience opportunity, which I was lucky enough to attend. For a couple of days, I sat with the Real-Time Indicators team. It was eye-opening to see how they use data from different sources — like energy prices, card transactions, and job adverts — to tell a story about the UK economy, week by week.

But I also noticed something: while the data was there, it wasn’t always easy to explore or understand. As a student, I found it hard to answer questions like “What happend with job adverts on a specific month in the past?” or “How has retail spending changed recently?” That’s when I came up with the idea for **ONS EconBot** — a simple chatbot that could act like a subject-matter expert and help users interact with real-time economic data using plain English.

The chatbot I built can take a question like “What’s happening with job adverts?” and return a relevant paragraph from past ONS bulletins, along with a rewritten summary in the same tone used by ONS reports. It also tracks sentiment over time and responds to questions like “What’s the mood of the economy on the basis of what we have published?” using real data and simple statistical tools. All of this is powered by open government data, sentence embeddings, and a lightweight language model.

At the same time, I continued working on my original question — whether job adverts actually *predict* employment trends. I gathered monthly job advert data and compared it with employment rate data in the East Midlands. I used basic time-series techniques like lag analysis and correlation to explore the relationship. The analysis showed that job adverts often *lead* changes in employment by one or two months, suggesting they might be useful as a short-term predictor.

This whole experience — from exploring data on my own, to visiting a national statistics office, to building a chatbot and analysing real trends — taught me a lot about how to ask meaningful questions and use data to find answers. It’s deepened my interest in economics, computing, and the role of public data in decision-making.

I didn’t set out to build a project that used AI or do something advanced. I started with a simple question about jobs. But by following that curiosity, I ended up learning tools like Python, natural language processing, and statistical analysis and LLM — all from scratch. This journey has made me excited about what’s possible when data is made open and accessible, and it’s confirmed that I want to continue studying computer science and data at a much deeper level.

## SOP Paragraph (with Technical Terminology)

My interest in computing and data science grew from a practical question I asked while exploring part-time jobs after completing my GCSEs: do increases in job adverts signal future improvements in employment? This led me to the Office for National Statistics’ Real-Time Indicators, where I began analysing monthly job advert volumes and regional employment rates for the East Midlands. I designed a time-series analysis using pandas and statsmodels to explore lagged correlations between job advert levels and employment rates, testing the hypothesis that labour demand acts as a short-term leading indicator. The results suggested a consistent one- to two-month lag, offering insight into the dynamics of local economic recovery.

While working with these datasets, I also developed a natural language assistant called **ONS EconBot**, which allows users to query UK economic indicators in plain English and receive structured, ONS-style responses. I used sentence-transformers (MiniLM) to embed bulletin paragraphs, indexed them with FAISS for semantic similarity search, and employed the flan-t5-base model to rewrite retrieved text into formal economic summaries. To assess changes in economic tone over time, I applied VADER sentiment analysis to paragraphs across multiple bulletins, generating a rolling sentiment tracker. The entire tool is built on open government data and open-source models and was designed with accessibility and public understanding in mind.

This project reflects how technical tools — from embedding models and vector search to sentiment scoring and lagged regression — can be combined to answer real-world questions. It deepened my interest in machine learning, information retrieval, and the intersection of statistics and public policy, and confirmed my motivation to pursue Computer Science at a university where these methods can be explored in far greater depth and rigour.