CS 210 : Data Management in Data Science Course Project Guidelines

General Instruction

You should work on a project that is related to the theme of the course: data management in data science. The scope of data management is broadly defined and can include any aspect of data collection, storage, integration, transformation, analysis, and visualization. I encourage you to pick a problem you are excited about and will be flexible if the project is relevant to topics and research papers in lectures.

There are two forms of final projects that you may choose from:

1. Research Project (Recommended)/Applied Project:
   - Formulate your own research question related to data management in data science and attempt to provide a solution, which can be partial with preliminary results. Your project can be computational, theoretical, experimental, or empirical. A research project can be done individually (not recommended) or by a group of 2 students (recommended, but no more than 2).

Extra Notes:

- Start Early: Start thinking about your project early and spend enough time developing it.

Deliverables

1. Project Proposal

Due: 23:59pm, June 20th, 2024 (encouraged to submit early)

Purpose: The written proposal should define your project/research question and explain what you are planning to do, so I can provide feedback on your proposal.

Length: 2 pages (no more than 3), typed, single-space.

Content:

- Define Project: What problem are you solving? What strategic aspects are involved? How does your project relate to the lectures/papers we discussed?

- Novelty and Importance: Why is your project important? Why are you excited about it? What are some existing issues in current data management practices? Are there any prior related works? Provide a brief summary.
- Plan: Be specific and succinct.
  - What kind of data will you use (if any)? How will you get it? Will you create or simulate it?
  - What models/techniques/algorithms do you plan to use or develop?
  - Do you have any hypothesis on your research question? How will you evaluate your method? How will you test and measure success?

2. Final Report
Due: 23:59pm, July 10th, 2024

Length: Normally, a well-explained project would take 6-8 pages, typed, single-space.

Content:
- Project Definition:
  - What problem are you solving? What strategic aspects are involved? How does your project relate to the lectures/papers we discussed?
  - Novelty and Importance:
    - Why is your project important? Why are you excited about it? What are some existing issues in current data management practices? Are there any prior related works? Provide a brief summary.
  - Depending on your individual case, the above two aspects can be an extended or revised version of what you have written in your proposal.
- Progress and Contribution:
  - What kind of data did you use (if any)? How did you get it?
  - What models/techniques/algorithms did you use or develop?
  - What experiments did you design?
  - What are the key findings or results from your project? Did they verify or refute your original hypothesis? How did you evaluate your method?
  - Discuss the advantages and limitations of your approach.
- Changes After Proposal:
  - If your final report differs from your proposed project, discuss the differences, why you made certain changes, and the bottlenecks that prevented you from proceeding with the proposed project.

Note to all: You may use tools to help with your writing but do not use generated contents directly. Please cite any tools, web sources, papers, and textbooks you consult/use. You are responsible for the content of your writing, including its originality and correctness. Plagiarism is not allowed.

Project Suggestions/Examples -

Customer Churn Prediction for a Telecom Company

Objective: Predict customer churn using historical customer data.

Steps and Technologies:

1. Data Collection:
    - Collect data from the company's CRM system and additional data via web scraping or APIs (e.g., customer reviews).
2. Data Storage:
    - Store collected data in a relational database (e.g., PostgreSQL).
    - Use a NoSQL database (e.g., MongoDB) for storing unstructured data such as customer reviews.
3. Data Integration:
    - Use ETL processes to integrate data from various sources.
    - Use Apache Airflow for workflow management.
4. Data Cleaning:
    - Handle missing values, outliers, and data inconsistencies using Pandas.
5. Data Transformation:
    - Feature engineering to create meaningful features.
    - Normalize and scale features.
6. Exploratory Data Analysis:
    - Visualize data trends and distributions using Matplotlib and Seaborn.
7. Model Building:
    - Build predictive models using machine learning algorithms (e.g., logistic regression, random forests) in Python.
8. Evaluation and Deployment:
    - Evaluate model performance using metrics like accuracy, precision, recall.
    - Deploy the model using Flask/Django for API creation.

2. Real-Time Sentiment Analysis on Social Media

Objective: Analyze and visualize the sentiment of tweets in real-time.

Steps and Technologies:

9. Data Collection:
    - Use Twitter API to collect tweets related to specific keywords.
10. Data Storage:
    - Store tweets in a NoSQL database (e.g., MongoDB).
11. Data Integration:
    - Use Apache Kafka for real-time data streaming.
    - Create an ETL pipeline to process incoming tweets.
12. Data Cleaning:
    - Clean tweets (remove URLs, mentions, hashtags, etc.) using regular expressions and NLTK.
13. Data Transformation:
    - Perform text preprocessing (tokenization, stop word removal, stemming/lemmatization).
14. Exploratory Data Analysis:
    - Visualize the frequency of words, hashtags, and sentiment distribution.
15. Sentiment Analysis:
    - Use NLP techniques and libraries (e.g., TextBlob, VADER) to classify tweets as positive, negative, or neutral.
16. Real-Time Dashboard:
    - Create a real-time dashboard using Plotly Dash or Streamlit to display sentiment analysis results.

3. Sales Forecasting for an E-commerce Platform

Objective: Forecast future sales based on historical sales data.

Steps and Technologies:

17. Data Collection:
    - Collect historical sales data from the e-commerce platform's database.
18. Data Storage:
    - Store data in a data warehouse (e.g., Amazon Redshift, Google BigQuery).
19. Data Integration:
    - Use Apache NiFi for data flow automation.
    - Set up ETL pipelines to load data into the data warehouse.
20. Data Cleaning:
    - Clean and preprocess data using Pandas and NumPy.
21. Data Transformation:
    - Perform feature engineering (e.g., create lag features, moving averages).
    - Handle seasonality and trends.
22. Exploratory Data Analysis:

- Visualize sales trends, seasonality, and other patterns.

23. Forecasting Models:
    - Build time series forecasting models (e.g., ARIMA, Prophet) using Python.
24. Model Evaluation and Deployment:
    - Evaluate model performance using metrics like MAE, RMSE.
    - Deploy the model using Flask/Django for API creation.
    - Create a user interface to display forecasts and historical data.

4. Healthcare Data Management and Analysis

Objective: Manage and analyze patient records to predict disease outbreaks and patient readmission rates.

Steps and Technologies:

25. Data Collection:
    - Collect patient records from hospital databases and external sources (e.g., public health databases).
26. Data Storage:
    - Use a combination of relational databases (e.g., MySQL) for structured data and NoSQL databases (e.g., Cassandra) for semi-structured data.
27. Data Integration:
    - Implement ETL processes using Apache Nifi or Talend.
    - Use Apache Airflow for orchestrating data workflows.
28. Data Cleaning:
    - Address data quality issues like missing values, duplicates, and inconsistencies.
29. Data Transformation:
    - Feature engineering (e.g., age group categorization, disease classification).
    - Normalize and scale data.
30. Exploratory Data Analysis:
    - Use data visualization tools to identify trends and anomalies.
31. Predictive Modeling:
    - Build predictive models for disease outbreak detection and readmission rate prediction using machine learning libraries (e.g., Scikit-learn).
32. Model Deployment:
    - Deploy models as APIs using Flask/Django.
    - Create a dashboard for real-time monitoring and visualization using Plotly Dash.

5. Building a Recommendation System for an Online Retailer

Objective: Develop a recommendation system to suggest products to customers based on their browsing and purchasing history.

Steps and Technologies:

33. Data Collection:
    - Gather browsing and purchase history data from the retailer's database.
    - Use web scraping and APIs to collect additional data on product reviews and ratings.
34. Data Storage:
    - Store data in a data lake (e.g., AWS S3, Azure Blob Storage).
35. Data Integration:
    - Use ETL processes to merge data from various sources.
    - Manage data pipelines with Apache Airflow.
36. Data Cleaning:
    - Clean and preprocess data using Pandas and NumPy.
37. Data Transformation:
    - Feature engineering to create user profiles and item vectors.
    - Normalize data and handle missing values.
38. Exploratory Data Analysis:
    - Visualize user behavior and product popularity.
39. Recommendation Algorithms:
    - Implement collaborative filtering, content-based filtering, and hybrid methods using Python.
40. Model Evaluation and Deployment:
    - Evaluate recommendation performance using metrics like precision, recall, and F1-score.
    - Deploy the recommendation system using Flask/Django.
    - Integrate with the retailer’s website for real-time recommendations.