knowledgehut upGrad

Data Science Bootcamp

Capstone Project

Guidebook

Introduction

Congratulations on successfully completing the data science program! Throughout this journey, you have acquired a wide range of data science skills, including data analysis, machine learning, and data visualization. Now, it's time to put all that learning into practice by working on your data science capstone project.

The data science capstone project is a pivotal part of the program and serves as a showcase of your abilities as a data scientist. This project is an opportunity for you to demonstrate your problem-solving skills, analytical thinking, and creativity in tackling real-world data challenges.

In this guide, we will provide you with essential points and guidelines for your data science capstone project. While we offer some project outlines, you are also encouraged to come up with your own unique project idea that aligns with your interests and showcases your skills effectively.

Completing this capstone project is a mandatory requirement to successfully graduate from the program, and it will be a gradable activity. So, make sure to approach it with enthusiasm and dedication.

If you have any questions or need guidance during the project development, don't hesitate to reach out to your buddy or learning advisor. We are here to support you throughout this journey.

Happy coding and best of luck with your data science capstone project!

The upGrad KnowledgeHut Team

Key Points to Consider Before Submitting Your Project:

When evaluating a data science capstone project, we look for the following attributes. It is essential for you to keep these in mind while building and submitting your data science capstone project:

1. **Data Exploration and Analysis:** Conduct thorough data exploration and analysis to gain insights into the data's characteristics, patterns, and relationships. Use appropriate visualizations and descriptive statistics to understand the data better.
2. **Data Preprocessing:** Clean, preprocess, and transform the data to make it suitable for modeling. Handle missing values, outliers, and data inconsistencies appropriately.
3. **Feature Engineering:** Create new features or transform existing features to improve model performance and capture relevant information from the data.
4. **Model Selection:** Choose appropriate machine learning algorithms or statistical models based on the problem type and data characteristics. Consider the trade-offs between interpretability and predictive power.
5. **Model Training and Evaluation:** Split the data into training and test sets and train the model on the training data. Evaluate the model's performance on the test data using relevant evaluation metrics.
6. **Model Interpretability:** Pay attention to model interpretability, especially if the project has real-world implications, use feature importance to explain model predictions.
7. **Documentation:** Provide clear and comprehensive documentation for your project. Describe the problem statement, data sources, data preprocessing steps, feature engineering, model selection, and evaluation metrics.
8. **Data Visualization:** Use effective data visualizations to communicate insights and model results. Visuals should be clear, informative, and visually appealing.
9. **Model Deployment:** Consider the deployment of the trained model in a real-world scenario if applicable. Discuss how the model can be integrated into an existing system or make predictions in real-time.
10. **Communication:** Clearly communicate your findings, methodology, and results in a way that is understandable to both technical and non-technical stakeholders.
11. **Bonus Points:** For bonus points, consider the following:

    - Package your solution in a well-structured zip file with a detailed README explaining how to set up and run the end-to-end pipeline.

    - Demonstrate excellent documentation skills by providing clear explanations of the models and their benefits to both stakeholders and potential users.

    - Include interactive components or visualizations in your project to enhance the user experience and facilitate understanding of the data and results.

Organizing Assets for Data Science Capstone Projects

Before you begin working on your data science capstone project, it is essential to organize your assets in a structured manner to facilitate smooth submission and evaluation. Follow these guidelines for organizing your project assets during development and deployment:

During Development: In this phase, organize your project files in the following folder structure:

project-folder: Name this folder to reflect your project's name in all lowercase, using kebab casing with no spaces in the name.

1. notebooks: This folder should contain Jupyter notebooks or Python scripts where you perform data exploration, preprocessing, feature engineering, and model building.
2. data: This folder should contain the dataset(s) used in the project. Include both raw and processed data, along with a README file explaining the dataset's attributes.
3. models: This folder should contain the trained machine learning models or statistical models used in the project. Include model serialization files (e.g., Pickle files) and any model artifacts.
4. visuals: This folder should contain data visualizations and plots generated during exploratory data analysis or model evaluation. Visualizations should be saved as image files (e.g., PNG or JPEG).
5. README.md: This Markdown file should include a detailed description of your project, problem statement, data sources, and explanations of your code and models. Also, provide instructions on how to run your code and reproduce the results.

Deployment: When setting up your project for deployment, follow these steps:

1. Package Dependencies: Ensure you have a requirements.txt file that lists all the Python dependencies required to run your project. This file is essential for installing the necessary libraries on the deployment server.
2. Model Serialization: Serialize your trained models into files that can be easily loaded and used for predictions. Save them in a specific folder (e.g., "models") within your project.
3. Data Preprocessing: Include any data preprocessing steps as separate functions or modules in your project. This ensures reproducibility during deployment.

Remember to update your README.md with instructions on how to access and interact with your deployed project. Providing clear and concise documentation ensures ease of evaluation and demonstrates your professionalism as a data scientist.

By organizing your data science capstone project assets effectively, you make it easier for evaluators to understand your work and appreciate the effort you put into building a compelling data-driven solution. Good luck with your data science capstone project!

Submitting the Data Science Project

Before submitting your data science capstone project, make sure to check the following points:

1. **Functionality:** Ensure that your data science project is working correctly and meets the defined objectives. All data preprocessing, modeling, and evaluation steps should be functional and produce valid results.
2. **Code Structure:** Organize your data science project code according to the advised folder structure. Keep all relevant files, notebooks, and scripts in appropriate directories to maintain a clear and logical project structure.
3. **Deployment:** If applicable, deploy your data science project to a cloud platform or hosting service to make it accessible for evaluation. The deployed application should be functional and demonstrate the intended data science solution.
4. **GitHub Repository:** Store your well-organized data science project code in a GitHub repository. Make sure to include all necessary files and folders, including data, notebooks, scripts, models, and visualizations.
5. **Readme.md:** Include a comprehensive Readme.md file in your GitHub repository. This document should provide clear instructions on setting up and running the data science project. Describe the project's purpose, problem statement, data sources, and methodology. Additionally, provide information on model training and evaluation, data preprocessing, and any necessary dependencies.

If you can confidently answer "Yes" to all the above points, you are ready to submit your data science capstone project. Submit the following to your learning advisor:

1. **GitHub Repo URL:** Provide the URL to your GitHub repository containing the well-organized and documented data science project code.
2. **Deployed Application URL:** If applicable, include the URL to the deployed application. Ensure that the application is publicly accessible for evaluation purposes.

What Should I Build?

Your data science capstone project is an opportunity to showcase your skills and abilities as a data scientist. It should reflect your expertise and demonstrate your problem-solving capabilities using data-driven approaches. Here are some guidelines and ideas to help you decide what to build for your data science capstone project:

1. **Choose a Relevant Problem:** Select a problem that is relevant and has practical applications in real-world scenarios. Look for challenges that involve data analysis, predictive modeling, recommendation systems, or any other data-related tasks that align with your interests.
2. **Real Data:** Whenever possible, work with real-world datasets. Real data provides unique challenges and insights that can enrich your learning experience and demonstrate your ability to handle real-world data.
3. **Focus on Depth and Quality:** Instead of tackling multiple small projects, focus on building a single project with depth and high-quality results. A comprehensive, well-executed project demonstrates your ability to work on complex data science problems.
4. **Innovative Approaches:** Consider using innovative data science techniques or methodologies in your project. Think beyond standard algorithms and explore cutting-edge research or unique solutions to stand out.
5. **Data Visualization:** Data visualization is a crucial aspect of data science projects. Showcase your skills by creating informative and visually appealing visualizations that communicate insights effectively.
6. **Ethical Considerations:** Be mindful of the ethical implications of your data science project. Ensure that you handle sensitive data responsibly and comply with privacy and security regulations.
7. **Industry Relevance:** If you have a specific industry or domain of interest, try to align your project with it. Demonstrating domain knowledge and expertise can be advantageous in certain job applications.
8. **Reproducibility:** Make your project reproducible. Provide clear instructions on how to set up the environment, reproduce the analysis, and obtain the same results.
9. **Documentation:** Maintain thorough documentation throughout your project. Explain your thought process, methodologies, data sources, and model evaluation in a clear and organized manner.

10. **Keep Learning:** Don't hesitate to learn new tools, libraries, or techniques while working on your project. Demonstrating a willingness to learn and adapt is highly valued by potential employers.
11. **Bring Your Own Ideas:** While the guide may provide some ideas, feel free to come up with your own project idea that excites you. Personal projects that reflect your interests and passion often lead to better outcomes.

Remember, the goal is not just to complete the program but to create an impressive and impactful data science capstone project that showcases your abilities and sets you apart from other candidates. Put in the time, effort, and attention to detail to create a project that you can be proud of and that leaves a lasting impression on potential employers.

Project 1

AnomaData (Automated Anomaly Detection for Predictive Maintenance)

Problem Statement:

Many different industries need predictive maintenance solutions to reduce risks and gain actionable insights through processing data from their equipment.

Although system failure is a very general issue that can occur in any machine, predicting the failure and taking steps to prevent such failure is most important for any machine or software application.

Predictive maintenance evaluates the condition of equipment by performing online monitoring. The goal is to perform maintenance before the equipment degrades or breaks down.

This Capstone project is aimed at predicting the machine breakdown by identifying the anomalies in the data.

The data we have contains about 18000+ rows collected over few days. The column 'y' contains the binary labels, with 1 denoting there is an anomaly. The rest of the columns are predictors.

Your focus in this exercise should be on the following:

The following is recommendation of the steps that should be employed towards attempting to solve this problem statement:

- **Exploratory Data Analysis:** Analyze and understand the data to identify patterns, relationships, and trends in the data by using Descriptive Statistics and Visualizations.
- **Data Cleaning:** This might include standardization, handling the missing values and outliers in the data.
- **Feature Engineering:** Create new features or transform the existing features for better performance of the ML Models.
- **Model Selection:** Choose the most appropriate model that can be used for this project.
- **Model Training:** Split the data into train & test sets and use the train set to estimate the best model parameters.
- **Model Validation:** Evaluate the performance of the model on data that was not used during the training process. The goal is to estimate the model's ability to generalize to new, unseen data and to identify any issues with the model, such as overfitting.
- **Model Deployment:** Model deployment is the process of making a trained machine learning model available for use in a production environment.

Timeline

We expect you to do your best and submit a solution within 2 weeks.

Deliverables

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline

Tasks/Activities List

Your code should contain the following activities/Analysis:

- Collect the time series data from the CSV file linked here.
- Exploratory Data Analysis (EDA) - Show the Data quality check, treat the missing values, outliers etc if any.
- Get the correct datatype for date.
- Feature Engineering and feature selection.
- Train/Test Split - Apply a sampling distribution to find the best split
- Choose the metrics for the model evaluation
- Model Selection, Training, Predicting and Assessment
- Hyperparameter Tuning/Model Improvement
- Model deployment plan.

Success Metrics

Below are the metrics for the successful submission of this case study.

- The accuracy of the model on the test data set should be > 75%(Subjective in nature)
- Add methods for Hyperparameter tuning.
- Perform model validation.

Bonus Points

- You can package your solution in a zip file included with a README that explains the installation and execution of the end-to-end pipeline.

- You can demonstrate your documentation skills by describing how it benefits our company.

Data

The dataset for this project can be accessed by clicking the link provided below.

Anoma_data.csv

Project 2

FindDefault (Prediction of Credit Card fraud)

Problem Statement:

A credit card is one of the most used financial products to make online purchases and payments. Though the Credit cards can be a convenient way to manage your finances, they can also be risky. Credit card fraud is the unauthorized use of someone else's credit card or credit card information to make purchases or withdraw cash.

It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.

The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

We have to build a classification model to predict whether a transaction is fraudulent or not.

Your focus in this project should be on the following:

The following is recommendation of the steps that should be employed towards attempting to solve this problem statement:

- **Exploratory Data Analysis:** Analyze and understand the data to identify patterns, relationships, and trends in the data by using Descriptive Statistics and Visualizations.
- **Data Cleaning:** This might include standardization, handling the missing values and outliers in the data.
- **Dealing with Imbalanced data:** This data set is highly imbalanced. The data should be balanced using the appropriate methods before moving onto model building.
- **Feature Engineering:** Create new features or transform the existing features for better performance of the ML Models.
- **Model Selection:** Choose the most appropriate model that can be used for this project.
- **Model Training:** Split the data into train & test sets and use the train set to estimate the best model parameters.
- **Model Validation:** Evaluate the performance of the model on data that was not used during the training process. The goal is to estimate the model's ability to generalize to new, unseen data and to identify any issues with the model, such as overfitting.
- **Model Deployment:** Model deployment is the process of making a trained machine learning model available for use in a production environment.

Timeline

We expect you to do your best and submit a solution within 2 weeks.

Deliverables

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline

Tasks/Activities List

Your code should contain the following activities/Analysis:

- Collect the time series data from the CSV file linked here.
- Exploratory Data Analysis (EDA) - Show the Data quality check, treat the missing values, outliers etc if any.
- Get the correct datatype for date.
- Balancing the data.
- Feature Engineering and feature selection.
- Train/Test Split - Apply a sampling distribution to find the best split.
- Choose the metrics for the model evaluation
- Model Selection, Training, Predicting and Assessment
- Hyperparameter Tuning/Model Improvement
- Model deployment plan.

Success Metrics

Below are the metrics for the successful submission of this case study.

- The accuracy of the model on the test data set should be > 75% (Subjective in nature)
- Add methods for Hyperparameter tuning.
- Perform model validation.

Bonus Points

- You can package your solution in a zip file included with a README that explains the installation and execution of the end-to-end pipeline.
- You can demonstrate your documentation skills by describing how it benefits our company.

Data:

The dataset for this project can be accessed by clicking the link provided below.

creditcard.csv

Project 3

DocAssist (Building Intelligent Medical Decision Support System)

Problem Statement

The objective of this project is to develop an intelligent medical decision support system that analyzes patient data to assist doctors in making informed decisions about the best treatment options for individual patients. By leveraging machine learning and data analysis, the system will provide personalized treatment recommendations based on the patient's medical history, symptoms, lab results, and other relevant factors.

Focus Areas:

- **Data Collection:** Gather patient data from electronic health records (EHRs), medical databases, and other relevant sources.
- **Data Preprocessing:** Clean, normalize, and handle missing values in the patient data to prepare it for analysis.
- **Feature Engineering:** Identify and extract meaningful features from the patient data, such as demographic information, medical history, diagnostic test results, and medication history.
- **Model Development:** Develop machine learning models to predict treatment outcomes based on patient data.
- **Treatment Recommendations:** Create an algorithm that generates treatment recommendations for individual patients based on the model predictions.
- **Model Interpretability:** Implement methods to interpret the model's predictions and provide insights to doctors.
- **User Interface:** Design an intuitive user interface for doctors to input patient data and receive treatment recommendations.

Deliverables:

- Data Collection and Preprocessing Pipeline
- Treatment Recommendation Algorithm
- Model Interpretability Report.
- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model

- Discussion of future work
- The source code used to create the pipeline.

Tasks/Activities List:

- **Data Collection:**
  - Gather patient data from electronic health records (EHRs) and medical databases.
  - Ensure data privacy and compliance with healthcare regulations.
- **Data Preprocessing:**
  - Clean and preprocess the patient data to handle missing values and remove noise.
  - Anonymize and encrypt sensitive patient information for security.
- **Feature Engineering:**
  - Extract relevant features from the patient data, such as demographics, medical history, and diagnostic results.
  - Transform categorical variables into numerical representations.
- **Model Development:**
  - Choose appropriate machine learning algorithms, such as logistic regression, random forests, or neural networks.
  - Train the models using the preprocessed patient data.
- **Model Interpretability:**
  - Implement methods such as feature importance
- **User Interface:**
  - Design a user-friendly interface that allows doctors to input patient data and receive treatment recommendations.
  - Ensure data security and confidentiality in the user interface.

Success Metrics:

- The predictive models should achieve high accuracy and precision in treatment outcome predictions.
- Doctors' feedback on the usefulness and effectiveness of the treatment recommendations.

Bonus Points:

- Implement explainable AI techniques to enhance the interpretability of the models.
- Incorporate patient feedback and outcome data to continuously improve treatment recommendations.
- Ensure compliance with healthcare regulations and data privacy laws throughout the project.

Data:

The dataset for this project can be accessed by clicking the link provided below.

dataset.csv

Project 4

For-rest from Fires (Fire Detection Model using Deep Learning)

Problem Statement:

Lately, there have been many fire outbreaks which is becoming a growing issue and the damage caused by these types of incidents is tremendous to nature and human interests. Such incidents have highlighted the need for more effective and efficient fire detection systems.

The traditional systems that rely on temperature or smoke sensors have limitations such as slow response time and inefficiency when the fire is at a distance from the detectors.

Moreover, these systems are also costly. As an alternative, researchers are exploring computer vision and image processing techniques as a cost-effective solution. One such method is the use of surveillance cameras to detect fires and alert the relevant parties.

Computer vision-based fire detection, which utilizes image processing techniques, has the potential to be a useful approach in situations where traditional methods are not feasible. The algorithm for fire detection uses visual characteristics of fires such as brightness, color, texture, flicker, and trembling edges to distinguish them from other stimuli.

This project is aimed at building a Fire Detection Model using Deep Learning.

Objectives:

The main goal of this project is to create a fire detection system. The key objectives of the project is to identify fires from the images we can get from surveillance system or other resources

Your focus in this project should be on the following:

The following is recommendation of the steps that should be employed towards attempting to solve this problem statement:

- **Data collection:** Collect data on different types of fires, including images and videos of flames, smoke, and heat. The data should be collected in a controlled environment to ensure that it is representative of real-world fires.
- **Data preprocessing**: Clean and preprocess the data to ensure that it is ready for use in training the deep learning model. This may include resizing images, normalizing pixel values, and splitting the data into training, validation, and test sets.
- **Model development:** Use deep learning techniques such as convolutional neural networks (CNNs) to develop a model that can detect fires in the collected data. The model should be trained on the preprocessed data, and the accuracy of the model should be evaluated using the test set.

Timeline

We expect you to do your best and submit a solution within 2 weeks.

Deliverables

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline

Tasks/Activities List

Your code should contain the following activities/Analysis:

- Collect the data from any possible resources.
- Data Preprocessing.
- Feature Engineering and feature selection.
- Train/Test Split
- Choose the metrics for the model evaluation
- Model Selection, Training, Predicting and Assessment
- Hyperparameter Tuning/Model Improvement
- Model deployment plan.

Success Metrics

Below are the metrics for the successful submission of this case study.

- The accuracy of the model on the test data set should be > 85%(Subjective in nature)
- Add methods for Hyperparameter tuning.
- Perform model validation.

Bonus Points

- You can package your solution in a zip file included with a README that explains the installation and execution of the end-to-end pipeline.

- You can demonstrate your documentation skills by describing how it benefits our company.

Data:

The dataset for this project can be accessed by clicking the link provided below.

FOREST_FIRE_SMOKE_AND_NON_FIRE_DATASET.zip

Project 5

MoodforMusic (An Intelligent Mood Detection and Music Recommendation Application)

Problem Statement:

The objective of this project is to build an application that detects the mood of users using still images or videos and recommends music accordingly. The system will use image or video analysis to infer the user's mood and provide personalized music recommendations to enhance their emotional experience.

Focus Areas:

- **Image/Video Analysis**: Develop algorithms to analyze still images or videos and extract mood-related features.
- **Mood Classification:** Create a machine learning model that classifies the user's mood based on the extracted features.
- **Music Recommendation:** Build a music recommendation engine that suggests music based on the user's mood classification.
- **User Interface:** Design an intuitive and user-friendly interface to capture images/videos and display music recommendations.

Deliverables:

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- User Interface
- Comprehensive Documentation
- Source Code
- Discussion of future work

Tasks/Activities List:

- **Data Collection:** Gather a diverse dataset of images/videos representing various moods.
- **Image/Video Preprocessing:** Clean, resize, and normalize the images/videos for analysis.

- **Feature Extraction:** Develop algorithms to extract mood-related features from the images/videos.
- **Mood Classification Model:** Choose and implement a suitable machine learning algorithm for mood classification.
- **Model Training:** Split the data into training and testing sets. Train the mood classification model.
- **Music Recommendation Engine:** Create a music recommendation system based on mood classifications.
- **User Interface Development:** Design and develop an interactive user interface for mood capture and music recommendations.
- **Integration:** Integrate the image/video analysis, mood classification, and music recommendation components.
- **Testing and Validation:** Evaluate the performance of the system using test data and user feedback.
- **Model Refinement:** Fine-tune the mood classification model and music recommendation engine for better accuracy and effectiveness.
- **Documentation:** Prepare comprehensive documentation explaining the entire project, including the technical aspects and usage instructions.
- **Deployment Plan:** Plan the deployment of the application on a web server or as a mobile app.

Success Metrics:

- **Mood Classification Accuracy:** The mood classification model should achieve an accuracy of >75% on the test dataset.
- **Music Recommendation Effectiveness:** Measure user satisfaction and engagement with the recommended music based on user feedback.
- **Deployment:** The application should be successfully deployed and accessible to users.

Bonus Points:

- Packaging and README: Package the solution with a detailed README explaining installation and execution of the end-to-end pipeline.
- Documentation Quality: Demonstrate strong documentation skills by explaining how the application benefits the users and the company.

Data:

The dataset for this project can be accessed by clicking the link provided below.

MoodforMusic.zip

Project 6

NutriGro (Building a Recommender System)

Problem Statement

The goal of this project is to develop a recommender system that provides product recommendations.

Your focus in this exercise should be on the following:

The following is recommendation of the steps that should be employed towards attempting to solve this problem statement:

- Data Collection: Gather historical order data from the database or e-commerce platform. Collect healthiness information through surveys or health records (if available).
- Data Preprocessing: Clean and preprocess the historical order and healthiness data, handle missing values, and encode categorical variables.
- Recommender System Development:
  - Choose and implement the appropriate recommender system algorithm (collaborative filtering, content-based, or hybrid).
  - Split the data into training and testing sets.
  - Train and fine-tune the recommender system ~~using historical order data~~.
- Feature Engineering:
  - Normalize or scale numerical features as necessary.
- Healthiness Prediction Model Development:
  - Choose a suitable machine learning/deep learning algorithm
  - Split the healthiness data into training and testing sets.
- Testing and Validation:
  - Evaluate the performance of the recommender system using appropriate metrics.
  - Conduct user testing and collect feedback for system improvements.
- Documentation and Deployment:
  - Prepare comprehensive documentation for the project, including the models' technical details and usage instructions.
  - Deploy the system on a web server or cloud platform for public use.

Timeline

We expect you to do your best and submit a solution within 2 weeks.

Deliverables

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline

Tasks/Activities List

Your code should contain the following activities/Analysis:

- Data Collection
- Data Preprocessing
- Feature Engineering
- Recommender System
- Evaluation
- Integration
- User Interface/API
- Testing and Validation
- Model Refinement
- Documentation
- Deployment Plan

Success Metrics

Below are the metrics for the successful submission of this case study.

- The accuracy of the model on the test data set should be > 75% (Subjective in nature)
- Add methods for Hyperparameter tuning.
- Perform model validation.

Bonus Points

- You can package your solution in a zip file included with a README that explains the installation and execution of the end-to-end pipeline.
- You can demonstrate your documentation skills by describing how it benefits our company.

Data

We construct a dataset via collecting data from Allrecipes.com. The website is chosen since it is one of the largest food-oriented social networks with 1.5 billion visits per year. In particular, we crawl 52,821 recipes from 27 categories posted between 2000 and 2018. For each recipe, we crawl its ingredients, image and the corresponding ratings from users. To ensure the quality of the data, we filter recipes that do not have images and that contain repeated ingredients and zero reviews. Then we obtain raw_data, including 1,160,267 users, 49,698 recipes with 38,131 ingredients and 3,794,003 interactions , which contains files as following:

- recipe_file:
  each recipe information is presented in one line, which consists of recipe id, name, average ratings of reviewers, image url, review nums, ingredients, cooking directions, nutritions, and reviews.
- interaction_file:
  each interaction is presented in one line, which consists of user id, recipe id, rating, and dateLastModified.
- images:
  49,698 images, named with recipe_id.jpg.

Besides, in order to evaluate recommendation models' performance, we holdout the latest 30% of interaction history to construct the test set, and split the remaining data into training (60%) and validation (10%) sets. Then we retain users which occur in both training and test sets, and obtain 68,768 users, 45,630 recipes with 33,147 ingredients and 1,093,845 interactions. We call this dataset as core_data, the dataset consists of files as following:

- recipe_file: each recipe information is presented in one line, which consists of recipe id, name, image url, ingredients, cooking directions, nutritions.
- interaction_file(train.rating/valid.rating/test.rating): each interaction is presented in one line, which consists of user id, recipe id, rating, and dateLastModified.
- images: 45,630 images, named with recipe_id.jpg.

The dataset for this project can be accessed by clicking the link provided below.

NutriGro.zip

Project 7

Propensify (Propensity Model to identify how likely certain target groups customers respond to the marketing campaign)

Problem Statement:

Are you aware of what, when, and why your customers will make a purchase? Many businesses undertake an intense pursuit to discover these answers, dedicating valuable resources to data-driven campaigns and high-cost strategies - yet the actual outcomes often remain elusive and disappointing.

Customer information is considered to be a valuable asset, however its true worth can only be established when it is used. Many companies have large collections of data that appear to be impressive, but upon further examination, they may contain outdated or unimportant information.

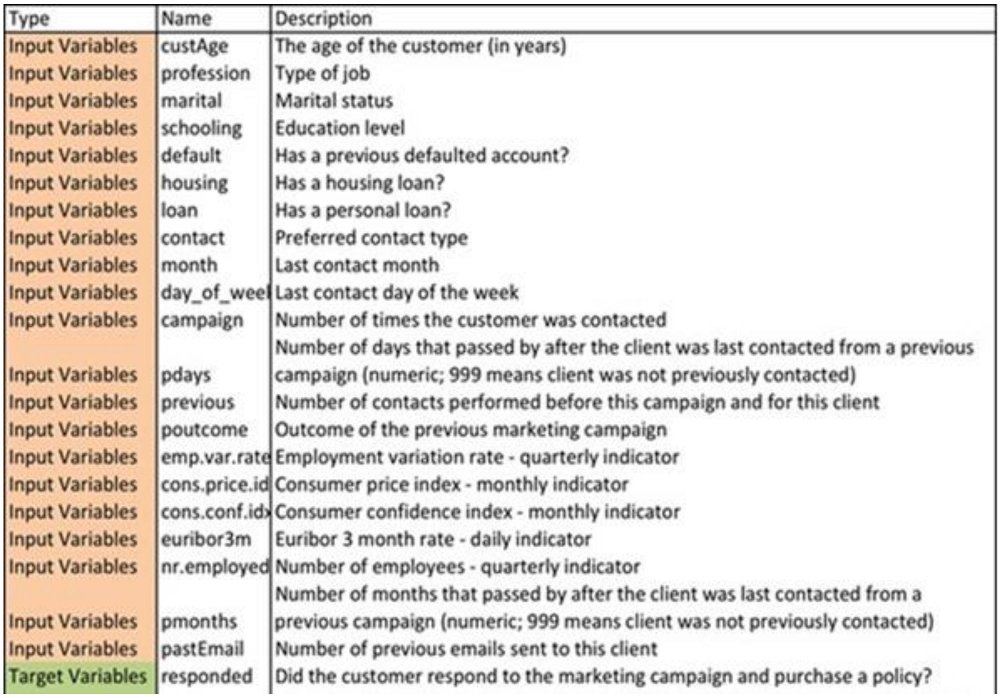
Propensity modeling is a method that aims to forecast the chance that individuals, leads, and customers will engage in specific actions. This method uses statistical analysis which takes into account all the independent and confounding factors that impact customer behavior.

Suppose you are working for a company as a Data Scientist. Your company is commissioned by an insurance company to develop a tool to optimize their marketing efforts.

This project is aimed at building a **propensity model to identify potential customers**.

Data:

The insurance company has provided you with a historical data set (train.csv). The company has also provided you with a list of potential customers to whom to market (test.csv). From this list of potential customers, you need to determine yes/no whether you wish to market to them. (Note: Ignore any additional columns available other than the listed below in the table)



Type	Name	Description
Input Variables	custAge	The age of the customer (in years)
Input Variables	profession	Type of job
Input Variables	marital	Marital status
Input Variables	schooling	Education level
Input Variables	default	Has a previous defaulted account?
Input Variables	housing	Has a housing loan?
Input Variables	loan	Has a personal loan?
Input Variables	contact	Preferred contact type
Input Variables	month	Last contact month
Input Variables	day_of_weel	Last contact day of the week
Input Variables	campaign	Number of times the customer was contacted
Input Variables	pdays	Number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
Input Variables	previous	Number of contacts performed before this campaign and for this client
Input Variables	poutcome	Outcome of the previous marketing campaign
Input Variables	emp.var.rate	Employment variation rate - quarterly indicator
Input Variables	cons.price.id	Consumer price index - monthly indicator
Input Variables	cons.conf.idx	Consumer confidence index - monthly indicator
Input Variables	euribor3m	Euribor 3 month rate - daily indicator
Input Variables	nr.employed	Number of employees - quarterly indicator
Input Variables	pmonths	Number of months that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
Input Variables	pastEmail	Number of previous emails sent to this client
Target Variables	responded	Did the customer respond to the marketing campaign and purchase a policy?

Your focus in this project should be on the following:

The following is recommendation of the steps that should be employed towards attempting to solve this problem statement:

- **Exploratory Data Analysis:** Analyze and understand the data to identify patterns, relationships, and trends in the data by using Descriptive Statistics and Visualizations.
- **Data Cleaning:** This might include standardization, handling the missing values and outliers in the data.
- **Dealing with Imbalanced data:** This data set is highly imbalanced. The data should be balanced using the appropriate methods before moving onto model building.
- **Feature Engineering:** Create new features or transform the existing features for better performance of the ML Models.
- **Model Selection:** Choose the most appropriate model that can be used for this project.
- **Model Training:** Split the data into train & test sets and use the train set to estimate the best model parameters.
- **Model Validation:** Evaluate the performance of the model on data that was not used during the training process. The goal is to estimate the model's ability to generalize to new, unseen data and to identify any issues with the model, such as overfitting.
- **Model Deployment:** Model deployment is the process of making a trained machine learning model available for use in a production environment.

Timeline

We expect you to do your best and submit a solution within 2 weeks.

Deliverables

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline

Tasks/Activities List

Your code should contain the following activities/Analysis:

- Collect the time series data from the CSV file linked here.
- Exploratory Data Analysis (EDA) - Show the Data quality check, treat the missing values, outliers etc if any.
- Get the correct datatype for date.
- Balancing the data.
- Feature Engineering and feature selection.
- Train/Test Split - Apply a sampling distribution to find the best split.
- Choose the metrics for the model evaluation
- Try multiple classification models and choose the best one.
- Model Selection, Training, Predicting and Assessment
- Hyperparameter Tuning/Model Improvement
- Please add a column to the testingCandidate.csv file. In this column, for each observation indicate a 1 (yes) or a 0 (no) whether you wish to market to that candidate.
- Model deployment plan.

Success Metrics

Below are the metrics for the successful submission of this case study.

- The accuracy of the model on the test data set should be > 85% (Subjective in nature)
- Add methods for Hyperparameter tuning.
- Perform model validation.

Bonus Points

- You can package your solution in a zip file included with a README that explains the installation and execution of the end-to-end pipeline.
- You can demonstrate your documentation skills by describing how it benefits our company.

Data

The dataset for this project can be accessed by clicking the link provided below.

Propensify.zip

Project 8

Recommender (Build intelligence to help customers discover products they may like and most likely purchase)

Problem Statement

In the e-commerce industry, providing personalized product recommendations to customers is crucial for enhancing user experience, increasing customer engagement, and boosting sales. Building an effective shopping recommender system can significantly improve customer satisfaction and drive repeat purchases. As a data scientist, your task is to develop a shopping recommender system that suggests relevant products to customers based on their preferences and browsing history, thereby increasing the likelihood of purchase and overall customer retention.

Project Focus: The main focus of this exercise is to build a shopping recommender system that can:

- Recommend products to customers based on their historical purchase data and browsing behavior.
- Enhance the customer shopping experience by providing personalized and relevant product suggestions.
- Increase customer engagement and satisfaction by accurately predicting products of interest.
- Leverage various recommendation techniques and evaluation metrics to deliver an optimal solution.

Timeline: The project is expected to be completed within one week.

Deliverables: Please submit the following deliverables in a zip file:

1. A comprehensive report (PDF) detailing:
    - Overview of the shopping recommender system's design and architecture.
    - Description of the recommendation techniques used (e.g., collaborative filtering, content-based filtering, hybrid models).
    - Evaluation metrics used to assess the recommender system's performance.
    - Explanation of how the system benefits the e-commerce organization and its customers.
2. Source code used to develop the shopping recommender system.
3. A user guide or README file describing how to install and run the end-to-end pipeline.

Tasks/Activities List: The project code should include the following activities and analyses:

1. Data Collection: Collect product data, customer profiles, purchase history, and browsing behavior from the e-commerce platform.
2. Data Preprocessing: Clean, preprocess, and transform the data to make it suitable for recommendation modeling.
3. Recommendation Techniques: Implement collaborative filtering, content-based filtering, and hybrid models for product recommendations.
4. Evaluation Metrics: Utilize metrics like precision, recall, F1-score, and mean average precision to evaluate the performance of the recommender system.

Success Metrics: The project's success will be evaluated based on the following metrics:

1. The recommender system should achieve a recommendation accuracy of 80% or higher on a validation dataset.
2. The system should provide personalized product recommendations to users based on their preferences and show high relevance to their interests.

Bonus Points: To earn bonus points, you can consider the following:

1. Package your solution in a well-structured zip file with a detailed README explaining how to set up and run the shopping recommender system.
2. Demonstrate excellent documentation skills by providing clear explanations of the models and their benefits to both customers and the e-commerce organization.

Data:

The dataset for this project can be accessed by clicking the icon provided below

Recommnder.zip

Project 9

FaultFindy (Build intelligence using Machine Learning to predict the faulty tyre in manufacturing)

The objective of this project is to develop an intelligent system using deep learning to predict the faults in manufacturing processes. By analyzing various manufacturing parameters and process data, the system will predict the faulty tyre generated during production. This predictive capability will enable manufacturers to proactively optimize their processes, reduce waste, and improve overall production efficiency.

Focus Areas:

- Data Collection: Gather historical manufacturing data, including good and faulty corresponding tyre images.
- Data Preprocessing: Clean, preprocess, and transform the data to make it suitable for deep learning models.
- Feature Engineering: Extract relevant features and identify key process variables that impact faulty tyre generation.
- Model Selection: Choose appropriate machine learning algorithms for faulty tyre prediction.
- Model Training: Train the selected models using the preprocessed data.
- Model Evaluation: Assess the performance of the trained models using appropriate evaluation metrics.
- Hyperparameter Tuning: Optimize model hyperparameters to improve predictive accuracy.

Deliverables:

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline.

Tasks/Activities List:

- **Data Collection:**
  - Gather historical manufacturing data, including good and faulty images.
  - Ensure data quality, handle missing values, and remove outliers.
- **Data Preprocessing:**
  - Clean and preprocess the data to remove noise and inconsistencies.
- **Feature Engineering:**
  - Identify important features and process variables that influence fault.
  - Engineer relevant features to capture patterns and correlations.
- **Model Selection:**
  - Choose appropriate machine and deep learning algorithms.
  - Consider models like logistic regression, decision trees, random forests, or gradient boosting, CNN, computer vision.
- **Model Training:**
  - Split the data into training and testing sets.
  - Train the selected machine learning models on the training data.
- **Model Evaluation:**
  - Evaluate the models' performance using relevant metrics
  - Choose the best-performing model for deployment.
- **Hyperparameter Tuning:**
  - Fine-tune hyperparameters of the selected model to optimize performance.
  - Use techniques like grid search or random search for hyperparameter optimization.

Success Metrics:

- The predictive model should achieve high accuracy

Bonus Points:

- Create visualizations or reports to communicate the model's predictions and insights to stakeholders.

- Implement a feedback loop to update the model periodically with fresh manufacturing data.

Data:

The dataset for this project can be accessed by clicking the link provided below

Faultfindy.zip

Project 10

Intensity Analysis (Build your own model using NLP and Python)

The objective of this project is to develop an intelligent system using NLP to predict the intensity in the text reviews. By analyzing various parameters and process data, the system will predict the intensity where its happiness, angriness or sadness. This predictive capability will enable to proactively optimize their processes, and improve overall customer satisfaction.

Focus Areas:

- Data Collection: Gather all the intensity data, including the text and its intensity.
- Data Preprocessing: Clean, preprocess, and transform the data to make it suitable for machine learning models.
- Feature Engineering: Extract relevant features and identify key process variables that impact intensity.
- Model Selection: Choose appropriate machine learning algorithms for intensity classification.
- Model Training: Train the selected models using the preprocessed data.
- Model Evaluation: Assess the performance of the trained models using appropriate evaluation metrics.
- Hyperparameter Tuning: Optimize model hyperparameters to improve predictive accuracy.
- Deployment: Deploy the trained model in a production environment for real-time predictions.

Your focus in this project should be on the following:

The following is recommendation of the steps that should be employed towards attempting to solve this problem statement:

- **Data Collection:** Collect and preprocess data for natural language processing (NLP) tasks. This may include text data for training the language model.
- **Model Development:** Use machine learning techniques such as deep learning to develop models for NLP. These models should be trained on the collected data and fine-tuned for improved accuracy.
- **Testing and Validation:** Test the classification in different scenarios and environments to ensure it can analyze appropriately. The performance of the model should be evaluated, and any issues should be addressed.

Timeline

We expect you to do your best and submit a solution within 2 weeks.

Deliverables

Please share the following deliverables in a zip file.

- A report (PDF) detailing:
- Description of design choices and Performance evaluation of the model
- Discussion of future work
- The source code used to create the pipeline

Tasks/Activities List

Your code should contain the following activities/Analysis:

- Collect the data from any possible resources.
- Data Preprocessing.
- Feature Engineering and feature selection.
- Train/Test Split
- Choose the metrics for the model evaluation
- Model Selection, Training, Predicting and Assessment
- Hyperparameter Tuning/Model Improvement
- Model deployment plan.

Success Metrics

Below are the metrics for the successful submission of this case study.

- The accuracy of the model on the test data set should be > 85% (Subjective in nature)
- Add methods for Hyperparameter tuning.
- Perform model validation.

Bonus Points

- You can package your solution in a zip file included with a README that explains the installation and execution of the end-to-end pipeline.
- You can demonstrate your documentation skills by describing how it benefits our company.

Data:

The dataset for this project can be accessed by clicking the link provided below

Intensity_data.zip