Salifort Motors: Employee Turnover Prediction Project

PACE Strategy Document - Construct stage

Google Advanced Data Analytics Capstone

Introduction

I will use this PACE strategy document to record my decisions and reflections as I move into the Construct stage of this capstone project. This document will guide my efforts to build models, apply feature engineering, and select appropriate evaluation metrics. It will help structure my approach to model development and support the integration of insights gained during earlier stages. Additionally, it will serve as a valuable reference to support my growth as a data professional, aiding in the construction of well-grounded and effective analytical solutions.

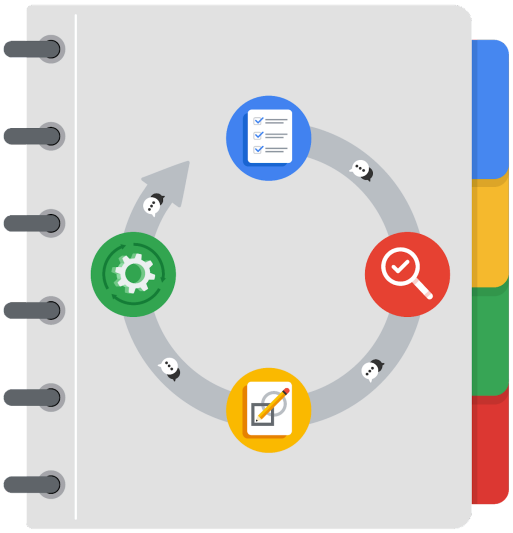
Portfolio Project Recap

Many of the goals I accomplished in my individual course portfolio projects are integrated into this Advanced Data Analytics capstone project. These include:

- Creating a clear and structured project proposal
- Demonstrating my understanding of Python's form and function
- Using Python to load, explore, extract, and organize information through custom functions
- Organizing and analyzing a dataset to uncover meaningful insights and tell the underlying "story"
- Developing a Jupyter notebook for exploratory data analysis (EDA)
- Computing descriptive statistics.
- Evaluating the model to assess its performance
- Applying machine learning techniques in a notebook environment to solve a defined problem
- Communicating results effectively by summarizing findings in an executive summary for external stakeholders

This capstone brings together all the skills I've developed across the program, allowing me to apply them in a cohesive, real-world project.

THE PACE WORKFLOW



[Alt-text: The PACE Workflow with the four stages in a circle: plan, analyze, construct, and execute.]

I will demonstrate to the company's HR team how I would apply the PACE workflow to the upcoming Salifort Motors project. For each question presented in this PACE strategy document, I will provide a structured and thoughtful response aligned with the corresponding stage of the PACE framework — Plan, Analyze, Construct, and Execute. This approach ensures clarity in my methodology, highlights my strategic planning skills, and illustrates a well-organized path from problem understanding to actionable insights.

Project Tasks

The following questions have been identified as essential to the Employee Turnover Prediction Project. These questions are primarily situated within the Construct stage of the PACE workflow, emphasizing the development and refinement of predictive models, feature engineering, and metric selection. I will address each question by aligning it with the most appropriate phase of the PACE framework — Plan, Analyze, Construct, or Execute — based on the specific project stage it reflects. To ensure each response is informed and well-reasoned, I will draw on relevant materials from the project notebook, the PACE framework, and best practices in model development and evaluation.

Data Project Questions & Considerations

PACE: Construct Stage

Get Started with Python

- Do any data variables averages look unusual?

The average monthly hours mean or average is 201.05 (~201 hours), which appears unusual. If I assume a standard 9–5 job with 8 working hours per day, then dividing 201 by 8 gives approximately 25 days of work per month. However, the typical number of working days in a month is around 20–23. This discrepancy suggests employees are working more than expected.

This elevated workload may be linked to attrition. A more grounded baseline—assuming a 40-hour work week and two weeks of vacation per year—puts the average monthly working hours for a full-time employee at approximately 166.67 hours (or roughly 166 hours). In comparison, the actual average of 201.05 hours is significantly higher.

As earlier analysis showed, apart from employees working on only two projects, every other group—regardless of whether they left or stayed—tended to work well above the theoretical full-time schedule. This strongly indicates that overwork could be a contributing factor to employee turnover.

- How many vendors, organizations or groupings are included in this total data?

The dataset includes several categorical groupings and variables representing different vendors, organizations, or groupings:

- Departments: There are 10 unique departments: sales, accounting, hr, technical, support, management, IT, product_mng, marketing, and RandD.

- Salary Levels: Salary is categorized into three groups — low, medium, and high.

- Number of Projects: This ranges from 2 to 7 projects.

- Time Spent in Company (Tenure): Ranges from 2 to 10 years.

- Work Accident: A binary variable indicating whether the employee experienced a work accident (0 = No, 1 = Yes).

- Attrition (Left): The outcome variable showing whether the employee left the company (0 = Stayed, 1 = Left).

These groupings provide a diverse set of features for understanding employee behavior, performance, and attrition patterns.

Go Beyond the Numbers: Translate Data into Insights

- What data visualizations, machine learning algorithms, or other data outputs will need to be built in order to complete the project goals?

To complete the project goals, the following data visualizations, machine learning algorithms, and data outputs need to be built:

Data Visualizations:

- Boxplots: Boxplots were used to visualize the distribution of data, assess skewness, and identify outliers. In some cases, a single variable was plotted, while in other cases, both x and y axes were used to explore relationships between multiple variables, with additional segmentation based on categorical features.

- Barplots: Useful for comparing categorical data such as employee departments or salary levels against attrition rates.

- Histograms: Helpful in understanding the distribution of numerical variables like satisfaction level, last evaluation, and average monthly hours.

- Scatterplots: Effective for exploring relationships between continuous variables, such as satisfaction level vs. average monthly hours.

- Grouped histograms: Ideal for comparing categorical variables across different groups, such as department and attrition.

- Stacked column charts: Useful for comparing the proportions of different categories within the data, like salary level and employee attrition status.

- Correlation heatmaps: To check for multicollinearity and also to get an idea of which fields affect the outcome variable, how strongly, and in which direction.

- Grouped column chart (Department vs Employee Count): This chart helps visualize the distribution of employees who stayed (0) versus those who left (1) across different departments.

- Line plots: To assess the linearity of the logit assumption in logistic regression.

- Confusion matrices: For evaluating classification models.

Model Evaluation Visuals:

- AUC-PR curve: To assess precision-recall tradeoff, especially given class imbalance.

- Decision tree splits: To visualize how decisions are made in tree-based models.

- Feature importance charts: For decision tree and random forest models to understand which variables contribute most to predictions.

Machine Learning Algorithms:

- Logistic Regression: As a baseline model and for interpretability.

- Decision Trees: With and without data leakage removal to understand overfitting risks.

- Random Forest: An ensemble model to improve performance and robustness, also tested with and without leakage removal.

Model Outputs:

- Cross-validation (CV) metric results: For model selection and validation.

- Test set metric results: For evaluating model generalizability.

These components together ensure robust data exploration, model building, evaluation, and interpretation aligned with the project's objectives.

- What processes need to be performed in order to build the necessary data visualizations?

To build the necessary data visualizations, the following processes need to be carried out:

- Import Visualization Libraries

Use seaborn and matplotlib—ensure the latest versions are installed for improved functionality and aesthetics.

- Understand the Data

Perform structural data checks using .info() and .describe() to assess completeness, data types, and basic statistical properties.

- Tailor Visuals to Audience

Design visualizations based on who will view them. For technical audiences, more detailed or complex visuals like correlation maps and model evaluation metrics are appropriate. For non-technical stakeholders, simpler visuals such as bar plots or stacked columns are more digestible.

- Maintain Aesthetic and Accessibility Standards

Use consistent color themes and avoid problematic color combinations for viewers with color vision deficiencies. Choose clear labels and readable fonts.

- Understand Variables Before Visualizing

It's important to understand what each field represents to avoid misleading visuals. For instance, boxplots might mislead if the sample size is small or uneven across groups.

- Key Visualizations to Include

- Boxplots: Useful for analyzing distribution, skewness, and spotting outliers in fields like satisfaction level and evaluation scores.

- Histograms: Show distribution of numerical variables such as satisfaction, evaluation, and monthly hours.

- Grouped histograms: Used to compare attrition rates across salary levels, departments, or tenure groups.

- Scatterplots: Show relationships between continuous variables, for example, satisfaction level vs. average monthly hours.

- Stacked Column Charts: Help visualize the proportion of employees who stayed or left within departments or job roles.

- Correlation Heatmaps: Identify relationships between variables and check for multicollinearity.

- Grouped column chart (Department vs Employee Count): This chart helps visualize the distribution of employees who stayed (0) versus those who left (1) across different departments.

- Line Plots: Inspect linearity of the logit for logistic regression models.

- Confusion Matrices: Show performance of classification models with true positives, true negatives, false positives, and false negatives.

- Decision Tree Diagrams: Visualize model splits for interpretability.

- Bar Charts for Feature Importance: Rank features based on importance from decision trees and random forests.

By carefully planning and customizing visualizations in this way, the analysis remains both effective and accessible to diverse audiences while supporting model development and decision-making.

- Which variables are most applicable for the visualizations in this data project?

The most applicable variables for visualizations in this data project include both numerical and categorical features, depending on the type of visualization and the use of hue.

Numerical Variables:

These are primarily used in boxplots, line plots, scatterplots, histograms, and correlation heatmaps:

- satisfaction_level

- last_evaluation

- number_project

- average_monthly_hours

- tenure

Categorical Variables:

These are especially relevant when hue is applied or when grouped visualizations are created:

- department

- left (used as hue for comparisons across attrition status)

- promotion_last_5years

- salary

Derived/Model-based Variables:

Used in model evaluation and feature importance plots:

- Logit (log-odds of `left`) — for assessing linearity in logistic regression

- Feature importance scores — from decision tree and random forest models

- Classification outcome — used in confusion matrices (True Positive, False Negative, etc.)

Hue-specific Variables:

Hue is used to enhance interpretation in many plots, and common hue variables include:

- left

- salary

- promotion_last_5years

- Classification outcome labels (e.g., TP, FP)

In summary, the visualizations in this project make use of a wide range of variables — with numerical features forming the foundation for distribution and relationship plots, and categorical variables driving grouped, hue-based, and class-based comparisons.

- Going back to the Plan stage, how do you plan to deal with the missing data (if any)?

In the current dataset, there are no missing values, so no imputation or removal was necessary. However, if missing data had been present, I would have addressed it through the following approaches based on the extent and pattern of the missingness:

- If the missing data is minimal, I would consider dropping the affected rows using:

  - `df.dropna(inplace=True)` or assigning the result back to the original DataFrame after `df.dropna()`.

- If missing values are more widespread or occur in important columns, I would use imputation methods:

  - Forward fill (`ffill`): Fills missing values by carrying forward the last known non-null value.

  - Backfill (`bfill`): Fills missing values using the next available non-null value.

  - Mean imputation: Replaces missing entries with the mean of the column.

  - Median imputation: Uses the column's median to fill in the missing values.

Missing values like geolocation data could be identified using `isna()` or similar checks. I could also create visualizations to understand which locations or categories are associated with higher missingness. This would help pinpoint where follow-up with the data provider might be needed.

Each of these strategies would be carefully selected based on the data distribution and the role of the affected column in analysis or modeling. The goal is to preserve data integrity while ensuring analytical reliability.

The Power of Statistics

- Is hypothesis testing necessary for this analysis? Additionally, what is the difference between the null hypothesis and the alternative hypothesis?

There was no dedicated hypothesis testing required in this project, as the project's goal and objective did not necessitate it.

Difference between Null Hypothesis and Alternative Hypothesis:

- Null Hypothesis (H_0): Assumes that there is no effect, no difference, or no relationship between variables. It represents the default or status quo condition.

- Alternative Hypothesis (H_1 or Ha): Proposes that there is an effect, a difference, or a relationship. It challenges the null hypothesis and suggests that a change has occurred.

In essence:

- H_0: Nothing has changed; any observed effect is due to random chance.

- H_1: Something has changed; the observed effect is statistically significant and not due to chance.

Regression Analysis: Simplify Complex Data Relationships

- Do you notice anything odd?

Yes, there are a few unexpected patterns in the data:

- In the correlation heatmap, some intuitive relationships didn't hold up as strongly as expected. For example, tenure and salary might naturally be assumed to correlate strongly, but the data shows only a weak relationship between them.

- In the salary-by-tenure plot, there's a surprising spike in employees with exactly 3 years of tenure, which could indicate either a hiring surge or some data irregularity.

- Can you improve it? Is there anything you would change about the model?

Yes, I believe the model can be improved further despite its strong performance. Here are some key areas for potential enhancement:

- Linearity and Feature Transformation: Several continuous variables did not meet the linearity assumption, which could affect coefficient interpretability. While this doesn't impact the model's classification performance, applying transformations (e.g., quadratic terms, splines) could improve robustness, especially if the focus shifts to inference.

- Increase Recall: Improving recall remains a priority to better capture leavers. This is critical for early identification, enabling more effective retention strategies.

- Feature Engineering & Data Leakage: Further feature engineering, such as aggregating or interacting features (e.g., job role, tenure, satisfaction), could uncover stronger signals. Additionally, while data leakage was previously checked, it's worth revisiting key features (e.g., last_evaluation) to assess their true impact on model performance.

- Model Tuning: Refining hyperparameters, adjusting cross-validation settings, experimenting with resampling techniques (e.g., SMOTE), and using class weights or threshold tuning could help optimize model performance, especially in handling class imbalance.

While this model is likely near-optimal given the extensive preprocessing already conducted, these steps could offer further improvements in predictive power and insight into employee attrition.

The Nuts and Bolts of Machine Learning

- Is there a problem? Can it be fixed? If so, how?

Yes, there was a problem of data leakage, which I observed during the modeling process. I identified it when certain features like `satisfaction_level` and `last_evaluation` appeared suspiciously optimistic, suggesting that the model might be learning from information not realistically available at prediction time.

Data leakage occurs when information that should not be available during training — either because it belongs to the test set or reflects future data — is inadvertently included in the model. This can lead to overly optimistic performance metrics that do not generalize well to unseen data and would underperform in real-world deployment.

In this case, it's unlikely that a company would consistently have up-to-date satisfaction scores or be able to use average monthly hours without bias. For example, if an employee is already planning to

leave, or if they've been flagged for termination, their working hours may already be reduced — effectively revealing the outcome we're trying to predict.

In the first round of modeling, I included all available features in the logistic regression, decision tree, and random forest models. In the second round, I addressed the data leakage by performing feature engineering:

- I dropped the `satisfaction_level` feature.

- I carried out feature extraction to create a new binary variable called `overworked`, derived from `average_monthly_hours`, which indicates whether an employee is working more than a typical amount.

This change helped reduce the risk of leakage while still preserving predictive value, allowing the model to generalize better to unseen scenarios.

- Which independent variables did you choose for the model, and why?

For the multiple binomial logistic regression model, I included all available features after appropriate encoding:

- I applied one-hot encoding to the `department` variable since it is nominal and has no inherent order.

- For the `salary` variable, I used ordinal encoding to reflect its natural rank (e.g., low < medium < high).

- All other variables were retained in their existing form and included as independent variables.

For the decision tree and random forest models (Round 1), I also used all features without exclusions.

However, in Round 2 of the decision tree and random forest models, I made a few adjustments:

- I dropped the `satisfaction_level` variable to explore how the model performs without it and to reduce the risk of overfitting.

- I engineered a new binary feature called `overworked`, derived from `average_monthly_hours`, to address potential data leakage. This new variable helped flag employees working significantly above average hours.

These adjusted features, alongside the rest of the dataset, were used as independent variables in the second round of modeling to test for improvements in interpretability and generalization.

- How well do your models fit the data? (What are the validation scores for each model?)

The models were evaluated based on ROC AUC scores for both cross-validation (CV) and test sets to assess how well they fit the data.

Decision Tree - Round 1 (hr_dt1)

- Cross-Validation (CV) Results:

ROC AUC: 0.9698

- Test Results:

ROC AUC: 0.9506

- Analysis:

The Decision Tree shows a strong fit, with a minor drop in test performance likely due to overfitting. Overall, it fits the data well.

Random Forest - Round 1 (hr_rf1)

- Cross-Validation (CV) Results:

ROC AUC: 0.9804

- Test Results:

ROC AUC: 0.9564

- Analysis:

The Random Forest performs excellently, with minimal drop in test results. It fits the data well and handles class imbalance effectively.

Decision Tree - Round 2 (hr_dt2)

- Cross-Validation (CV) Results:

ROC AUC: 0.9587

- Test Results:

ROC AUC: 0.9336

- Analysis:

The second Decision Tree shows some overfitting, with a more noticeable drop in test performance. It indicates a less optimal fit compared to the first round.

Random Forest - Round 2 (hr_rf2) – Champion Model

- Cross-Validation (CV) Results:

ROC AUC: 0.9648

- Test Results:

ROC AUC: 0.9384

- Analysis:

The Round 2 Random Forest provides the best performance with strong consistency between training and test sets. This makes it the best fit for the data and the chosen champion model.

In conclusion, the Round 2 Random Forest model performed consistently well on both the training and test sets, showing strong generalization, and was selected as the champion model for further use.

- Can you improve it? Is there anything you would change about the model?

Yes, I believe the model can still be improved in several ways, even though it performs well overall. Here are some areas that could lead to better performance:

- Increase Recall: Since recall is a critical metric for identifying leavers, improving it would help capture more at-risk employees. This is especially important in a business context, where detecting leavers early can lead to better retention strategies.

- Feature Engineering: There might be additional features or transformations of existing features that could better capture the patterns associated with employee turnover. For example, aggregating or interacting features like job role, tenure, or satisfaction levels might provide stronger signals.

- Investigate Data Leakage: Even though prior checks were done, it's important to continually evaluate potential data leakage. Features like `last_evaluation` and `satisfaction_level` may reflect outcomes rather than the causes of attrition. I could assess model performance with and without these features to determine their actual contribution.

- Reframe the Problem: If certain features, like `last_evaluation`, strongly correlate with attrition, it might make sense to reframe the problem. For instance, I could try predicting satisfaction or evaluation scores themselves, which could act as leading indicators of turnover, and then use those as inputs for attrition prediction.

- Model Tuning:

- Adjust the cross-validation (`cv`) parameters during hyperparameter searches to ensure the model is as robust as possible.

- Experiment with resampling techniques like SMOTE or undersampling to handle class imbalance and improve the model's ability to detect the minority class (leavers).

- Consider using class weights or threshold tuning to optimize the model based on business priorities, focusing on reducing false negatives (i.e., failing to detect leavers).

Ultimately, it's possible that this model represents near-optimal performance with the current dataset, considering the extensive preprocessing and data quality work already done. However, implementing the steps mentioned above could help unlock further improvements and provide deeper insights into the factors contributing to employee attrition.

- Do you have any ethical considerations in this stage?

Yes, at this stage, I will focus on minimizing potential bias to ensure that the model produces fair and equitable predictions. My goal is to develop a baseline model that is technically robust and also aligned with the overall problem-solving objective. I'll pay close attention to selecting and evaluating the most relevant performance metrics for the scenario to minimize the risk of misclassification and unintended consequences. Throughout this phase, fairness, transparency, and reliability will remain core priorities.