Additional Support - Linear Regression

**Additional Support**

This page provides guidance on common challenges, best practices, and additional resources for selected steps in the assignment.

**Data Preprocessing and Feature Engineering**

**Fixing Datatypes**

**Why separate categorical features from numerical ones?**

* Machine learning models treat categorical and numerical features differently. Categorical variables often require encoding (one-hot or label encoding), while numerical features can be used directly in computations.
* Keeping them separate initially ensures correct transformations without unintended numerical operations on categorical data.

**Train-Validation Split**

**Common questions:**

* **Why split the data before feature engineering?**  
  Prevents data leakage, ensuring that the validation set remains unseen during preprocessing.
* **What’s a good split ratio?**  
  80-20 or 70-30 is common. Ensure that the distribution of target and key features remains consistent across both sets.

**Exploratory Data Analysis (EDA)**

**Why perform EDA separately on training and testing data sets?**

* **Data Leakage:** If EDA is performed on the entire dataset (both training and testing), information from the test set may influence decisions made during model building. This leads to overfitting, where the model performs well on the training data but poorly on new, unseen data.
* **Accurate Performance Evaluation:** By performing EDA separately on the test set, we can ensure that the test data is distributed well enough to represent the real-world patterns the model might encounter. Well-distributed test data also helps to get an accurate evaluation of the model’s generalisation abilities.

**Feature Distribution**

**Understanding skewness and transformations:**

* Skewness indicates asymmetry in data distribution. High skewness can impact model assumptions.
* Transformations like log or square root normalisation can help make data more normally distributed.

**Resources:**

* [Understanding Skewness in Data](https://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm)
* [Transformation in Python](https://opendatascience.com/transforming-skewed-data-for-machine-learning/)

**Correlation Analysis**

**Why drop weakly correlated features?**

Linear regression assigns coefficients based on relationships, so why do we want to drop any of the features when the model can simply assign small coefficients to less useful features?

* Weakly correlated features introduce noise and reduce model interpretability.
* Removing them reduces dimensionality and prevents overfitting.
* Feature selection ensures a more stable model by focusing on impactful predictors.

**Resources:**

* [Correlation Heatmap Tutorial](https://seaborn.pydata.org/generated/seaborn.heatmap.html)

**Model Building**

**Feature Scaling**

**Why scale when linear regression is scale-agnostic?**

* While linear regression does not require feature scaling for estimating coefficients, applying scaling can improve numerical stability and make the results easier to interpret.
* Scaling also helps gradient-based optimisation methods converge more quickly and enables a better comparison of feature importance, as standardised coefficients reflect their relative impact more clearly.
* For linear regression models, however, most libraries prefer the Ordinary Least Squares (OLS) method.

**Effect on the final model:**

* Unscaled features with large values might dominate model interpretation.
* When you read about advanced regression techniques, you will see that scaling ensures better convergence in gradient-based methods like Ridge and Lasso.

**Resources:**

* [When and Why to Scale Data](https://towardsai.net/p/data-science/how-when-and-why-should-you-normalize-standardize-rescale-your-data-3f083def38ff)
* [Effect of Scaling on Linear Regression](https://datascience.stackexchange.com/questions/30302/linear-regression-and-scaling-of-data)

**Libraries for Linear Regression**

* statsmodels: Provides detailed statistical summaries, including confidence intervals and p-values. Good for interpretability.
* sklearn: Optimised for predictive performance but lacks statistical insights.

**Recursive Feature Elimination (RFE)**

**How to select features using RFE?**

* RFE ranks features by recursively removing least important ones and refitting the model.
* Iterate over different feature subsets and compare performance metrics (R-squared, adjusted R-squared). This helps in deciding an optimal number of features that justify the model performance.

**Manual vs RFE-based feature selection:**

* **Manual:** Based on domain knowledge and exploratory analysis.
* **RFE:** Algorithmic, identifies features that contribute the most to prediction accuracy.

**Resources:**

* [Feature Selection with RFE](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html)

**Results and Inference**

**Residual Analysis**

**Interpreting residual plots:**

* **Homoscedasticity:** Residuals should have constant variance; patterns indicate heteroscedasticity.
* **Normality:** Use QQ-plots to check if residuals are normally distributed.
* **Independence:** Ensure residuals are uncorrelated.

**Resources:**

* [How to Interpret Residual Plots](https://online.stat.psu.edu/stat462/node/117/)
* [Assumptions of Linear Regression](https://online.stat.psu.edu/stat500/lesson/9/9.2/9.2.3)
* [QQ-Plots in Python](https://www.statsmodels.org/dev/generated/statsmodels.graphics.gofplots.qqplot.html)

**Coefficient Analysis**

**How scaling affects coefficients?**

* In standard-scaled models, coefficients represent the change in output per unit standard deviation of the feature. In other kinds of scaling, the scale of change varies as per the scaling function used.
* Convert back to the original scale by multiplying with the feature scaling factor.
* Use standardised coefficients to compare feature importance.

**Calculating the effect of unit change in scaled features:**

An example to understand how standard scaling affects the predicted output:  
If scaled\_coeff = β and original feature std = σ, the real-world impact is β \* σ per unit change in original scale.

| **Field** | **Description** |
| --- | --- |
| market\_id | Integer ID representing the market where the restaurant is located. |
| created\_at | Timestamp when the order was placed. |
| actual\_delivery\_time | Timestamp when the order was delivered. |
| store\_primary\_category | Category of the restaurant (e.g., fast food, dine-in). |
| order\_protocol | Integer representing how the order was placed (e.g., via Porter, call to restaurant, etc.). |
| total\_items | Total number of items in the order. |
| subtotal | Final price of the order. |
| num\_distinct\_items | Number of distinct items in the order. |
| min\_item\_price | Price of the cheapest item in the order. |
| max\_item\_price | Price of the most expensive item in the order. |
| total\_onshift\_dashers | Number of delivery partners on duty when the order was placed. |
| total\_busy\_dashers | Number of delivery partners already occupied with other orders. |
| total\_outstanding\_orders | Number of orders pending fulfillment at the time of the order. |
| distance | Total distance from the restaurant to the customer. |

Next, you will find the final submission field for this assignment.

**Submission Guidelines**

1. To submit your solution, push your submission to ***GitHub***and submit the ***GitHub*link**below.
2. **You are required to upload your solution as a zip file** titled  
   ***"LR\_Delivery\_Time\_Prediction\_<your\_name>.zip"***in a **public GitHub repository**.
3. The repository should be named after the assignment.
4. **The zipped folder that you upload should contain two files**:
   1. an **Interactive Python Notebook** (*.ipynb*) that contains your code, and
   2. a**Report Document** (*.pdf*) that presents your visualizations, analysis, results, insights, and outcomes.
5. Note that these files should only be generated from the starter files provided to you.
6. Both your Jupyter notebook and report should contain your name and the assignment title.
7. Mention all assumptions made in the report.
8. Your answers to all the tasks mentioned in the starter notebook should be present in the report.
9. Any graphs/plots you generate for analysis should also be attached to the report.