# Taxi Price Regression

## UE - Machine Learning

M2 - DS2E
VASTA Francesca
FRACASSI Diego
FREZARD Paul
VU Billy

# SUMMARY

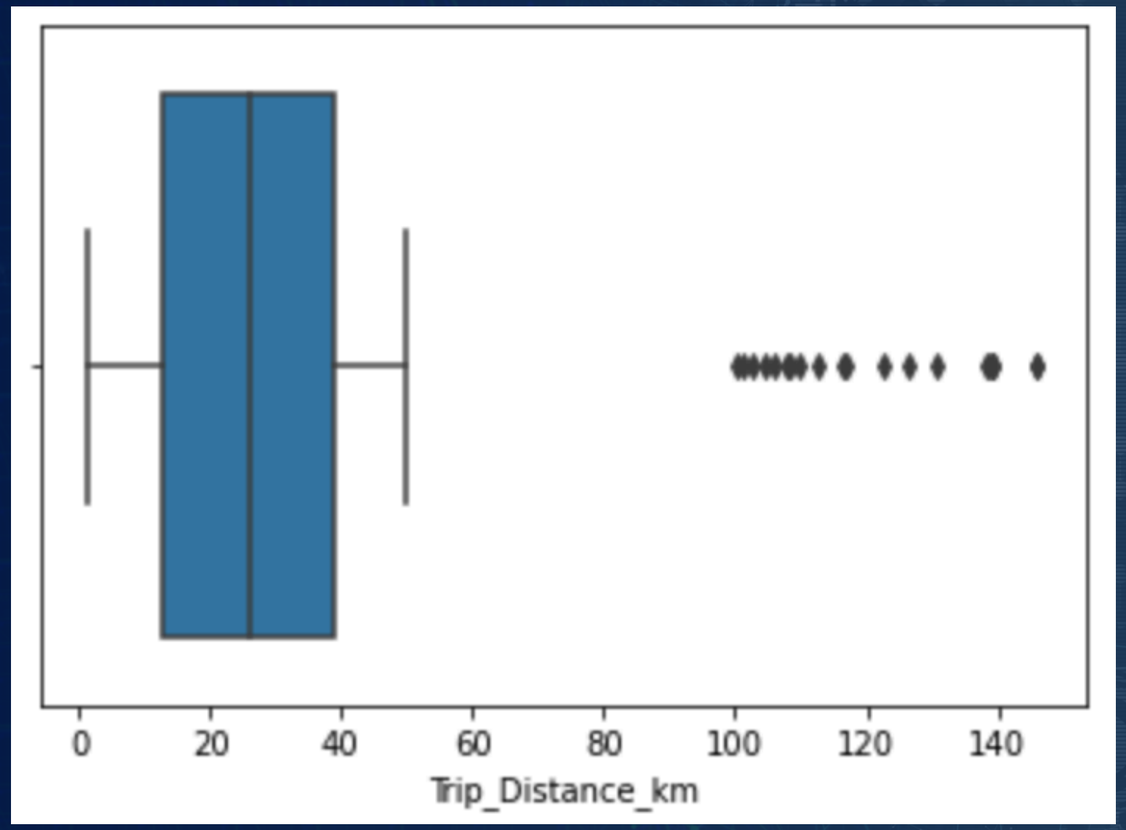
# INTRODUCTION AND GOAL

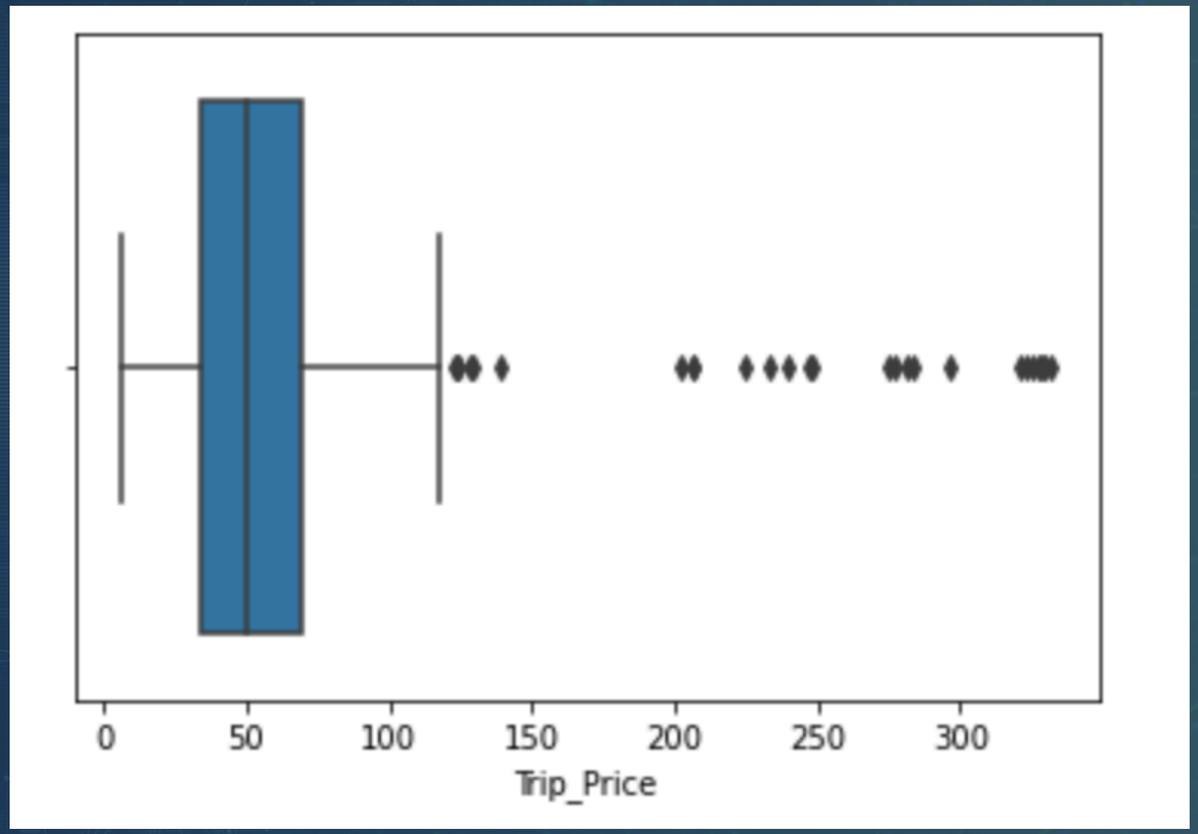
This dataset is designed to predict taxi trip fares based on various factors such as distance, time of day, traffic conditions, and more. It provides realistic synthetic data for regression tasks, offering a unique opportunity to explore pricing trends in the taxi industry.

# DATASET

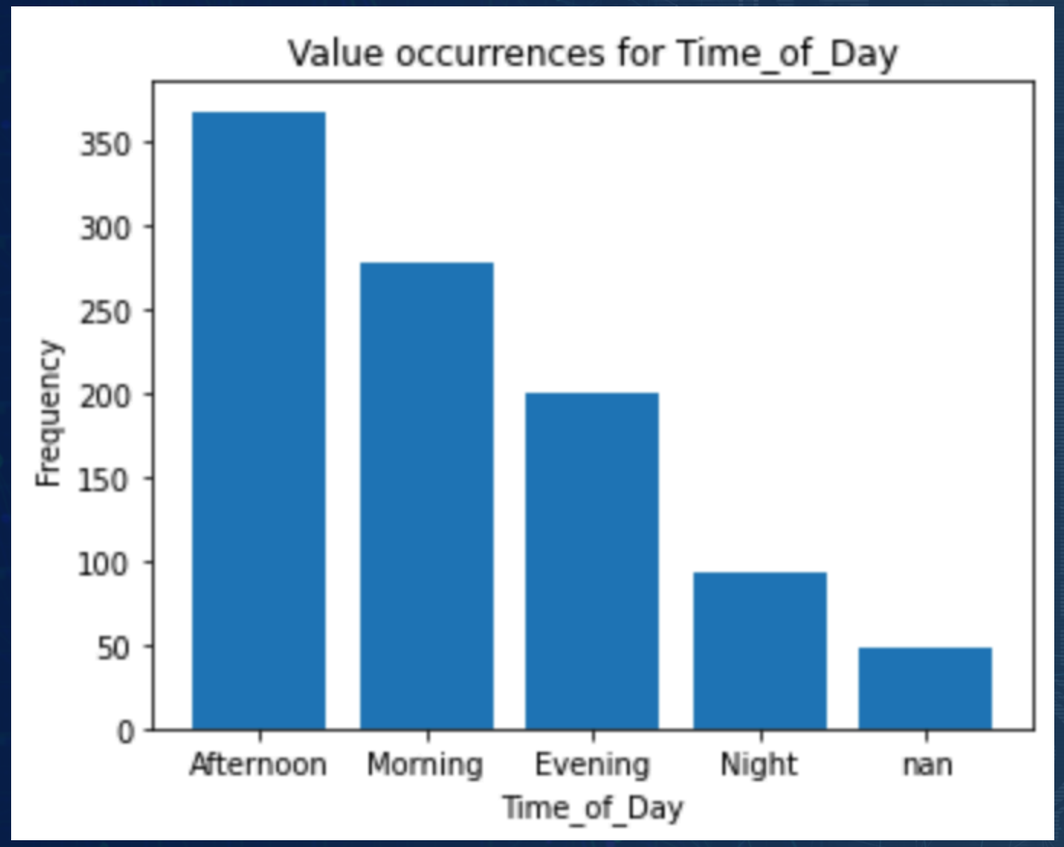
- 1000 observations
- 11 Features : 7 numerical / 4 categorial
  - Trip_Distance_km (numerical)
  - Time_of_Day (categorial)
  - Day_of_Week (categorial)
  - Passengen_Count (numerical)
  - Traffic_Conditions (categorial)
  - Weather (categorial)
  - Base_Fare (numerical)
  - Per_Km_Rate (numerical)
  - Per_Minute_Rate (numerical)
  - Trip_Duration_Minutes (numerical)
  - Trip_Price(numerical)

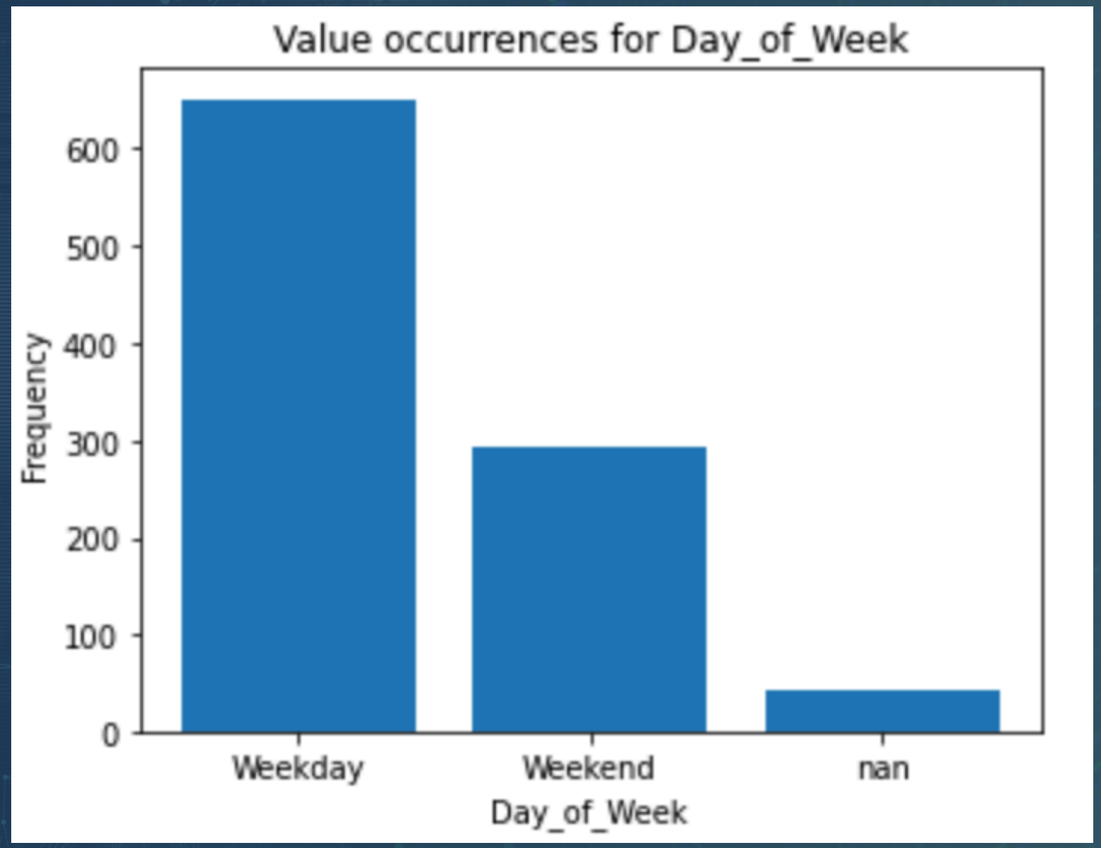
# EXPLORATORY DATA ANALYSIS (EDA)



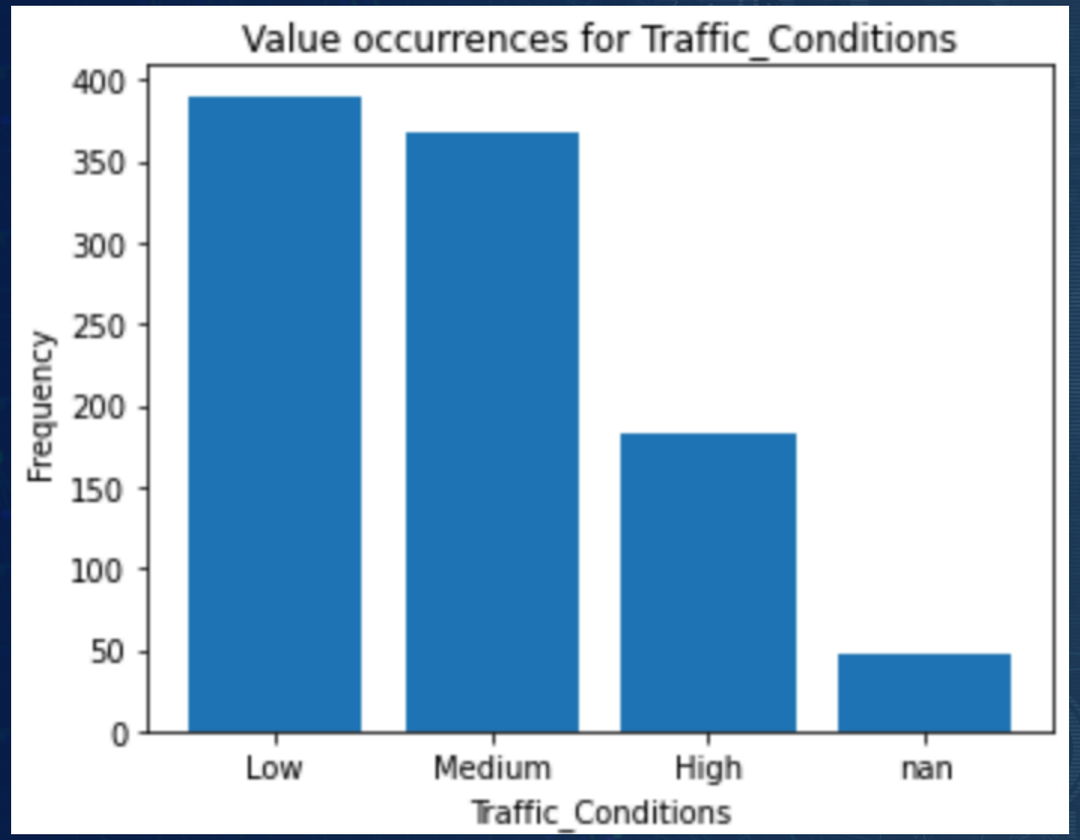



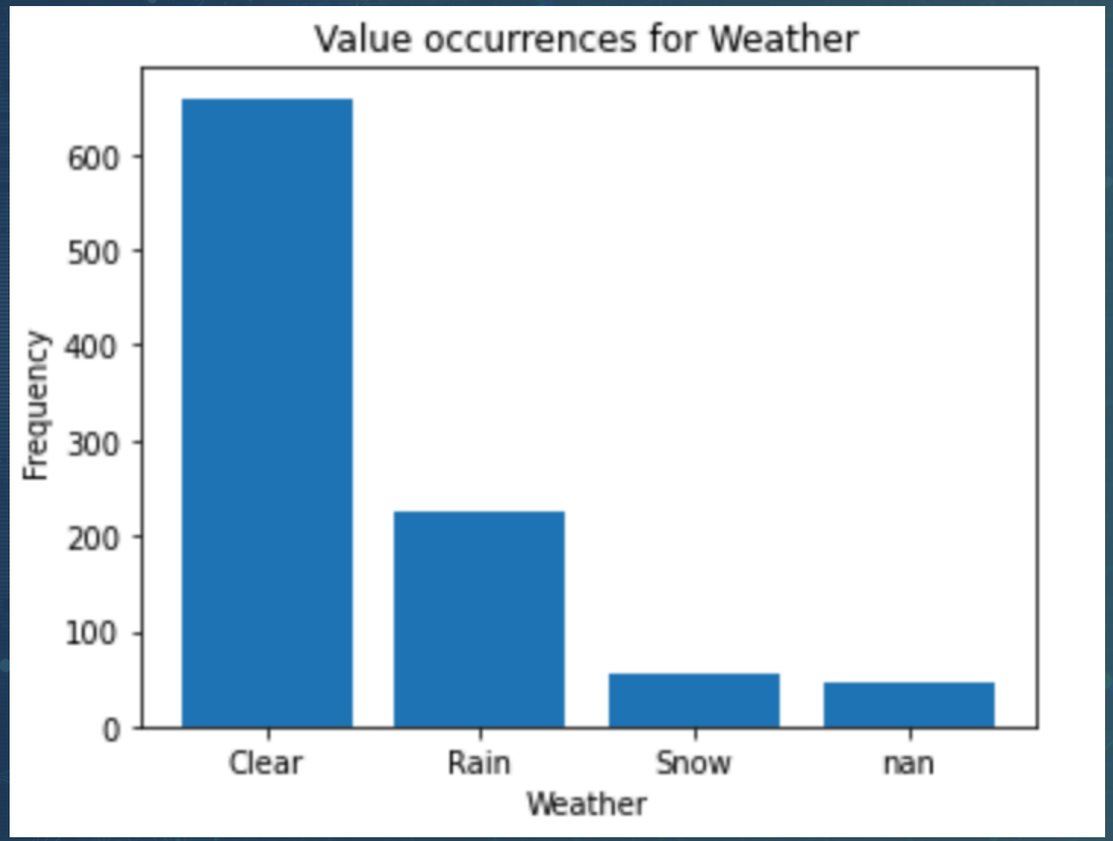
# EXPLORATORY DATA ANALYSIS (EDA)



# EXPLORATORY DATA ANALYSIS (EDA)



# FEATURE ENGINEERING

## NAs

**Counted the distribution of NAs for each variable**
:

- Trip_Distance_km : 47
- Time_of_Day : 49
- Day_of_Week : 43
- Passengen_Count : 44
- Traffic_Conditions : 47
- Weather : 46
- Base_Fare : 48
- Per_Km_Rate : 47
- Per_Minute_Rate : 47
- Trip_Duration_Minutes : 45
- Trip_Price : 44

**... and  for each row**

- 0 NA : 562 rows
- 1 NA : 341 rows
- 2 NA : 83 rows
- 3 NA : 14 rows

**We by immediately dropped the rows with more than 2 NAs.**
**For all the other observations we filled NAs using different techniques**

# FEATURE ENGINEERING

## Filling NAs for Numerical Variables

**How did we fill the NA ?**

**Average, mode and some mathematical formulas**

| | |
|---|---|
| **Numerical continous** | Base_Fare -> Average |
| **Numerical discrete** | Passengers_Count -> Mode |
| **Numerical continous and linked by mathematically** | ['Per_Minute_Rate'] = (['Trip_Price'] - ['Base_Fare' ]- ['Per_Km_Rate']*['Trip_Distance_km'] ) / ['Trip_Duration_Minutes']<br>['Per_Km_Rate'] = (['Trip_Price']- ['Base_Fare'] - ['Per_Minute_Rate'] * ['Trip_Duration_Minutes']) / ['Trip_Distance_km']<br>['Trip_Duration_Minutes'] = (['Trip_Price' ]- ['Base_Fare'] - ['Per_Km_Rate']*['Trip_Distance_km']) / ['Per_Minute_Rate']<br>['Trip_Distance_km'] = (['Trip_Price' - ['Base_Fare'] - ['Per_Minute_Rate'] * ['Trip_Duration_Minutes'])/ ['Per_Km_Rate']<br>['Trip_Price'] = ['Per_Minute_Rate'] * ['Trip_Duration_Minutes'] + ['Per_Km_Rate']*['Trip_Distance_km'] + ['Base_Fare'] |

# FEATURE ENGINEERING

## Filling NAs for Categorical Variables

Traffic_Conditions, Day_of_Week, Time_of_Day, Weather

**How did we fill the NA ?**

**KNN classifier**

**The goal: predict labels using only quantitative variables**

**Training set**

**Dataset without NAs**

All quantitative variables included
+ one categorical variable at a time

**Test set**

**Dataset with NAs**

All quantitative variables included + only
one categorical variable at a time  with NAs

# FEATURE ENGINEERING

## Final Dataset

```
RangeIndex: 891 entries, 0 to 890
Data columns (total 13 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Trip_Distance_km       891 non-null    float64
 1   Time_of_Day            891 non-null    object
 2   Day_of_Week            891 non-null    object
 3   Passenger_Count        891 non-null    float64
 4   Traffic_Conditions     891 non-null    object
 5   Weather                891 non-null    object
 6   Base_Fare              891 non-null    float64
 7   Per_Km_Rate            891 non-null    float64
 8   Per_Minute_Rate        891 non-null    float64
 9   Trip_Duration_Minutes  891 non-null    float64
 10  Trip_Price             891 non-null    float64
 11  Trip_Distance_km_log   891 non-null    float64
 12  Trip_Price_log         891 non-null    float64
dtypes: float64(9), object(4)
```

# MODELISATION

## Linear Models

1. Ordinary Least Squares
2. Best Subset Selection
3. Elastic Net Regularization

## Support Vector Regression

1. SVR with parameters found by Cross Validation

Mean Squared Error (test and training)
R^2 / Adjusted R^2

## Decision Tree Regression

1. Decision Tree
2. Cost-Complexity pruning
3. Random Forests
4. Gradient Boosting

Accuracy on training set
Accuracy on test set
Impurity at nodes

# LINEAR MODELS

## Ordinary Least Squares

OLS before outliers removal

```
Mean Squared Error: 0.1056
R-squared: 0.9112
```

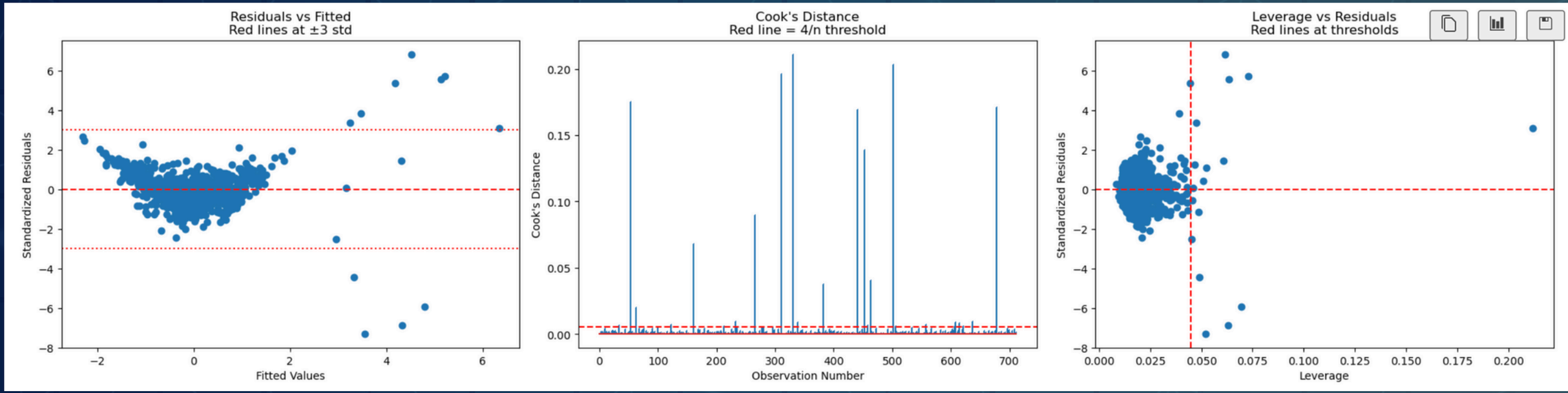
OLS after outliers removal

```
Mean Squared Error: 0.0818
R-squared: 0.9060
```

Removing outliers improved a bit the model accuracy

# OUTLIERS

Identification though classic diagnostic plots of fitted values vs. residuals (standardized), Cook's distance and leverage plots.

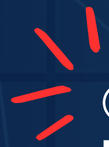



The red lines represent the arbitrary thresholds we set to identify the most influential residuals, which should be excluded from the training model.

# OUTLIERS

**Since the exclusion of outliers was not really beneficial for the Linear Regression Model, we decided to keep these influential observations in the dataset. (in total 31)**
**Moreover, Support Vector Machines (SVMs) and Decision Trees are not particularly sensitive to outliers in any case.**

↓



Of course, we acknowledge that **outlier exclusion depends a lot on the context of the analysis.**

# LINEAR MODELS

## Ordinary Least Squares

### MODEL SUMMARY

```
Mean Squared Error: 0.1056
R-squared: 0.9112
```

Accuracy = 1 - MSE = 0.8944= 89.44%

| | coef | std err | t | P>\|t\| |
|---|---|---|---|---|
| const | -0.0355 | 0.031 | -1.128 | 0.260 |
| Trip_Distance_km | 0.8023 | 0.012 | 69.568 | 0.000 |
| Passenger_Count | 0.0064 | 0.011 | 0.566 | 0.571 |
| Base_Fare | 0.0195 | 0.011 | 1.714 | 0.087 |
| Per_Km_Rate | 0.3611 | 0.011 | 31.510 | 0.000 |
| Per_Minute_Rate | 0.2566 | 0.011 | 22.996 | 0.000 |
| Trip_Duration_Minutes | 0.2790 | 0.011 | 24.473 | 0.000 |
| Time_of_Day_Evening | 0.0370 | 0.030 | 1.224 | 0.221 |
| Time_of_Day_Morning | 0.0353 | 0.028 | 1.275 | 0.203 |
| Time_of_Day_Night | -0.0076 | 0.041 | -0.186 | 0.852 |
| Day_of_Week_Weekend | 0.0054 | 0.024 | 0.220 | 0.826 |
| Traffic_Conditions_Low | 0.0162 | 0.032 | 0.512 | 0.609 |
| Traffic_Conditions_Medium | 0.0040 | 0.032 | 0.125 | 0.901 |
| Weather_Rain | 0.0157 | 0.026 | 0.596 | 0.551 |
| Weather_Snow | 0.0147 | 0.049 | 0.302 | 0.763 |

Only 5 predictors are significant

# LINEAR MODELS

## Best Subset Selection

**MODEL SUMMARY**
**(based on highest Adjusted R^2) )**

```
Adjusted R²: 0.7708
```

```
=== Performance on Test Set ===
Mean Squared Error: 0.1057
R-squared: 0.8974
```

**Accuracy = 1 - MSE = 0.8943 = 89.43%**

| | Feature | Coefficient |
|---|---|---|
| 0 | Intercept | -0.006090 |
| 1 | Trip_Distance_km | 0.802613 |
| 2 | Passenger_Count | 0.082238 |
| 3 | Base_Fare | -0.017063 |
| 4 | Per_Km_Rate | 0.254800 |
| 5 | Per_Minute_Rate | 0.106132 |
| 6 | Trip_Duration_Minutes | 0.178032 |
| 7 | Day_of_Week_Weekend | 0.057731 |
| 8 | Weather_Rain | -0.058676 |

**we can't see the p-values, but the best model includes 8 predictors, including dummies**

# LINEAR MODELS

## Regularized Regression

### ELASTIC NET MODEL SUMMARY

**Optimal parameters found by CV**

```
Best alpha: 0.011016263283795158
Best L1 Ratio (Mix Lasso-Ridge): 0.5
Intercept (β0): -0.0031
```

```
Mean Squared Error(test): 0.1048
R-squared: 0.8983
```

**Accuracy = 1 - MSE = 0.8952 = 89.52%**

| | Feature | Coefficient |
|---|---|---|
| 0 | Trip_Distance_km | 0.790599 |
| 1 | Passenger_Count | 0.001986 |
| 2 | Base_Fare | 0.012914 |
| 3 | Per_Km_Rate | 0.355051 |
| 4 | Per_Minute_Rate | 0.249703 |
| 5 | Trip_Duration_Minutes | 0.272150 |
| 6 | Time_of_Day_Evening | 0.000000 |
| 7 | Time_of_Day_Morning | 0.000000 |
| 8 | Time_of_Day_Night | -0.000000 |
| 9 | Day_of_Week_Weekend | 0.000000 |
| 10 | Traffic_Conditions_Low | 0.000000 |
| 11 | Traffic_Conditions_Medium | -0.000000 |
| 12 | Weather_Rain | 0.000000 |
| 13 | Weather_Snow | 0.000000 |

# LINEAR MODELS

Adding layers to the analysis with **Interaction Terms**

## OLS Regression with Interaction Terms

- Trip Distance (km) and Weekday
- Trip Duration (minutes) and Time of the Day
- Rate per km and Traffic Conditions

### MODEL SUMMARY

```
Mean Squared Error(test): 0.1076
R-squared: 0.9121
```

**Accuracy = 1 - MSE = 0.8924 = 89.24%**

| | coef | std err | t | P>\|t\| |
|---|---|---|---|---|
| const | -0.0186 | 0.018 | -1.042 | 0.298 |
| Trip_Distance_km | 0.7959 | 0.013 | 59.271 | 0.000 |
| Passenger_Count | 0.0064 | 0.011 | 0.568 | 0.571 |
| Base_Fare | 0.0180 | 0.011 | 1.587 | 0.113 |
| Per_Km_Rate | 0.4259 | 0.027 | 15.904 | 0.000 |
| Per_Minute_Rate | 0.2583 | 0.011 | 23.271 | 0.000 |
| Trip_Duration_Minutes | 0.2792 | 0.014 | 20.539 | 0.000 |
| Time_of_Day_Evening | 0.0320 | 0.030 | 1.063 | 0.288 |
| Time_of_Day_Morning | 0.0343 | 0.028 | 1.244 | 0.214 |
| Time_of_Day_Night | -0.0117 | 0.040 | -0.290 | 0.772 |
| Distance_Weekend_Interaction | 0.0223 | 0.025 | 0.895 | 0.371 |
| KmRate_TrafficMedium_Interaction | -0.0827 | 0.032 | -2.553 | 0.011 |
| KmRate_TrafficLow_Interaction | -0.0749 | 0.032 | -2.359 | 0.019 |
| Duration_Snow_Interaction | 0.0320 | 0.054 | 0.591 | 0.555 |
| Duration_Rain_Interaction | -0.0051 | 0.026 | -0.191 | 0.848 |

**2 interaction terms now have got a significant impact on the dependent variable**

# LINEAR MODELS

Ranking of best models

1. **Best Subset Selection** — 89,83%
2. **Regularized Regression** — 89,52%
3. **Ordinary Least Squares** — 89,44%
4. **OLS Regression with Interaction Terms** — 89,24%

# INTERPRETATION OF RESULTS

**Best Subset Selection**

| | |
|---|---|
| Intercept | -0.021899 |
| Trip_Distance_km | 0.503311 |
| Base_Fare | 0.000516 |
| Per_Km_Rate | 0.004884 |
| Per_Minute_Rate | 0.001025 |
| Trip_Duration_Minutes | 0.280973 |
| Time_of_Day_Evening | 0.037822 |
| Time_of_Day_Morning | 0.036118 |

For better interpretation of the results, we have converted the coefficients back to their original scale.

beta_original = beta_standardized * std_X / std_y

Each additional kilometer increases the trip price by approximately €0.50

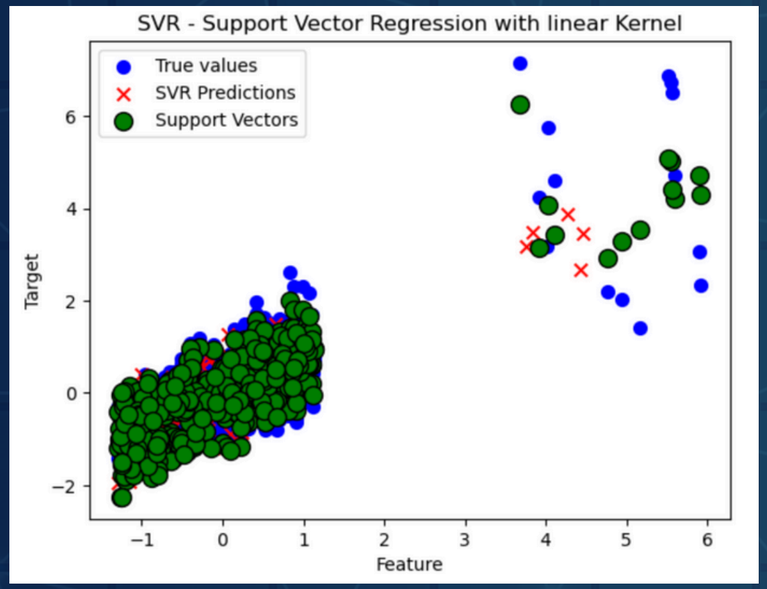
# SUPPORT VECTOR REGRESSION

## Support Vector Regression with linear kernel

**Model Summary**

```
Mean Squared Error: 0.1068
R-squared: 0.8964
```

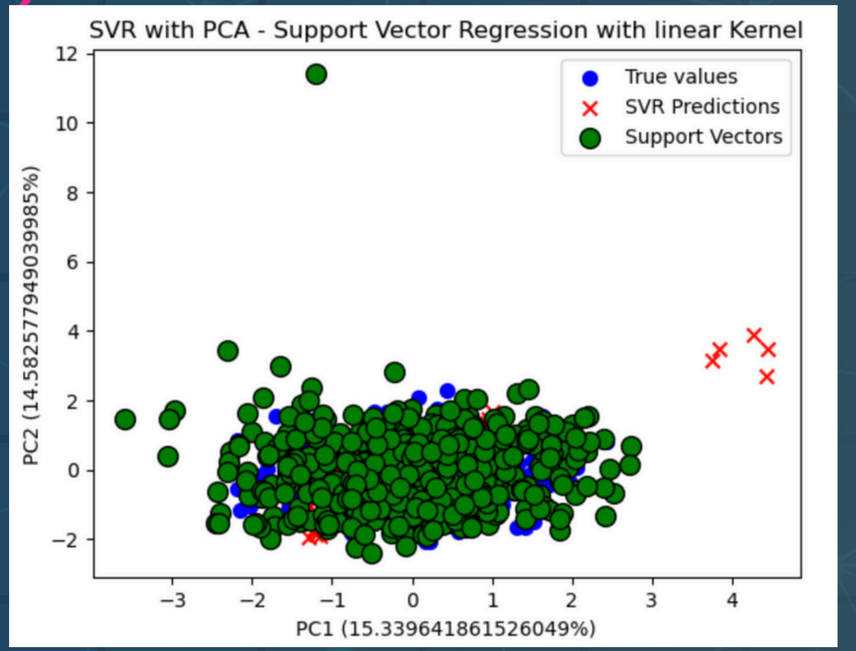
**Accuracy = 1 - MSE = 0.8932 = 89.32%**




**How many support vectors?**

```
Support Vectors: (446, 14)
```

Risk of overfitting



# SUPPORT VECTOR REGRESSION

## Support Vector Regression with CV

**Best parameters found by cross validation**

- Regularization parameter (C): **100**
- Influence of each single data point on the model (gamma): **0.01**
- Margin of tolerance of predictions (epsilon): **0.1**
- Kernel function: '**rbf**' (Radial Basis Function)

```
Mean Squared Error: 0.0054
R-squared: 0.9948
```

**Accuracy = 1 - MSE = 0.9946 = 99.46%**

# SUPPORT VECTOR REGRESSION

**Support Vector Regression**

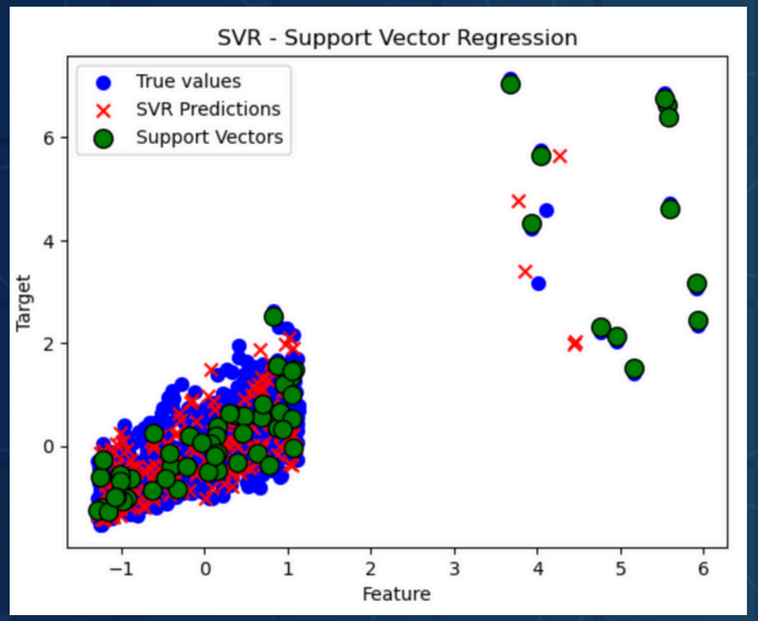
**How many support vectors?**

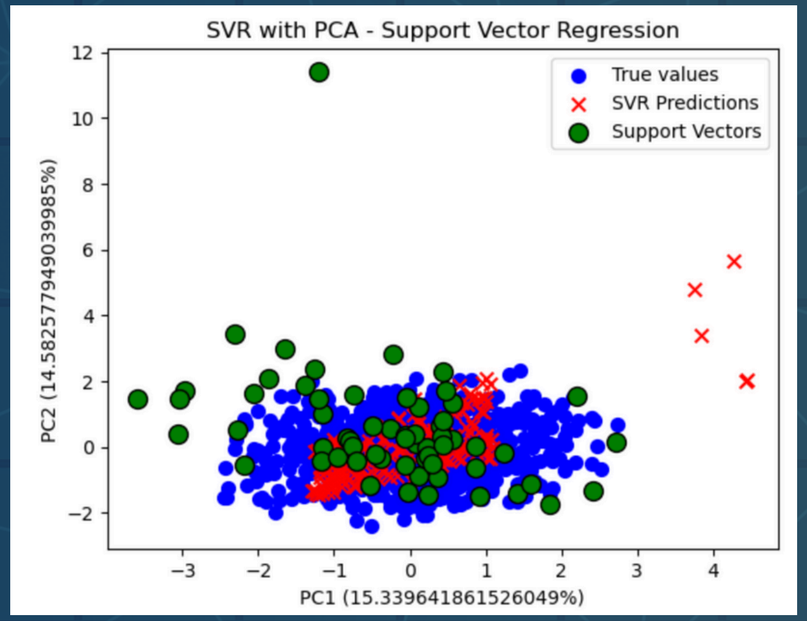
```
Support Vectors Shape: (303, 14)
```

There are only 62 support vectors, each one with 14 features

```
Support Vectors: (62, 14)
```

**NO MORE risk of overfitting**



# DECISION TREES

**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Decision trees

**NO PARAMETERS**

```
Accuracy on training set: 1.000
Accuracy on test set: 0.894
```

```
Depth of the tree:  17
Number of nodes in the tree:  712
Number of features used:  10
```

```
Average Impurity of the Tree nodes: 0.0126
Average Impurity of Leaf Nodes: -0.0000
```

| | Feature | Importance |
|---|---|---|
| 0 | Trip_Distance_km | 0.655976 |
| 7 | Per_Km_Rate | 0.194492 |
| 9 | Trip_Duration_Minutes | 0.079263 |
| 8 | Per_Minute_Rate | 0.059643 |
| 6 | Base_Fare | 0.006105 |
| 1 | Time_of_Day | 0.001705 |
| 3 | Passenger_Count | 0.001400 |
| 2 | Day_of_Week | 0.000694 |
| 4 | Traffic_Conditions | 0.000656 |
| 5 | Weather | 0.000065 |

# DECISION TREES

**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Decision trees

### FIXED MAX DEPTH (=4)

```
Accuracy on training set: 0.842
Accuracy on test set: 0.833
```

```
Depth of the tree:  4
Number of nodes in the tree:  16
Number of features used:  10
```

```
Average Impurity of the Tree nodes: 0.3321
Average Impurity of Leaf Nodes: 0.0860
```

| | Feature | Importance |
|---|---|---|
| 0 | Trip_Distance_km | 0.740811 |
| 7 | Per_Km_Rate | 0.176736 |
| 9 | Trip_Duration_Minutes | 0.055880 |
| 8 | Per_Minute_Rate | 0.022776 |
| 1 | Time_of_Day | 0.002837 |
| 6 | Base_Fare | 0.000961 |
| 2 | Day_of_Week | 0.000000 |
| 3 | Passenger_Count | 0.000000 |
| 4 | Traffic_Conditions | 0.000000 |
| 5 | Weather | 0.000000 |

# DECISION TREES

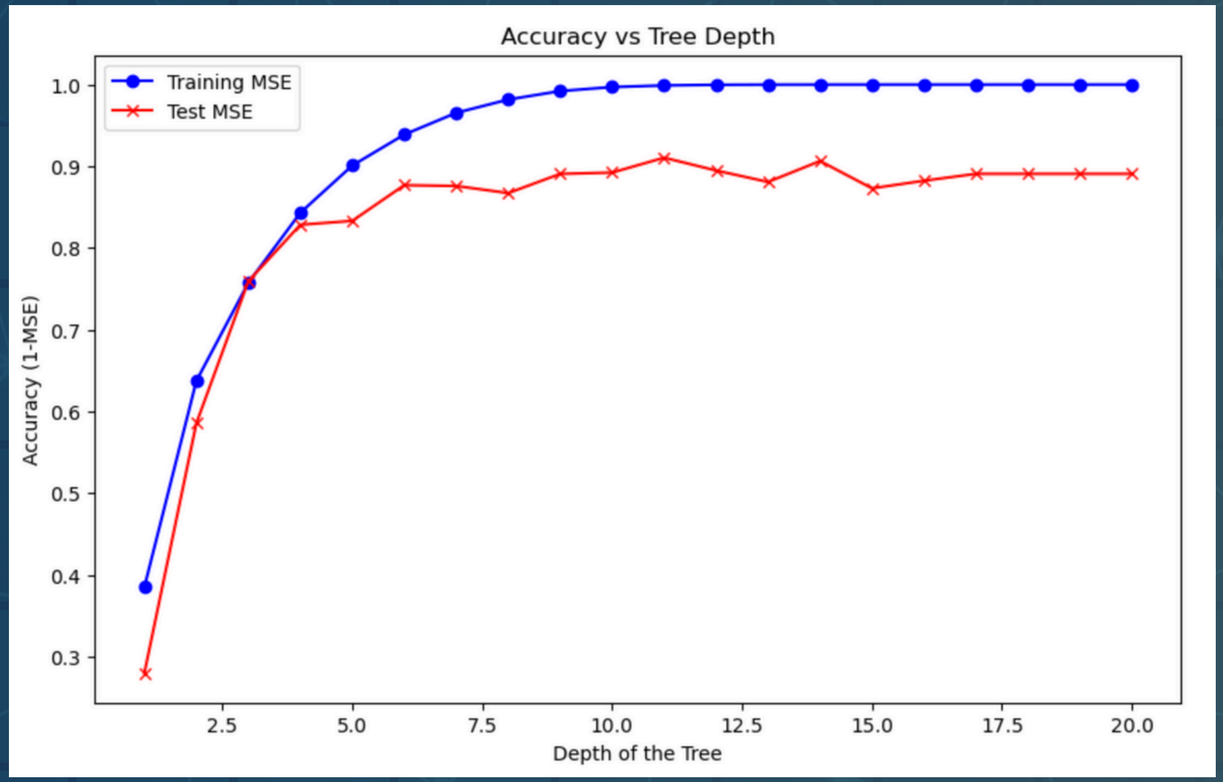
**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Decision trees

### Impact of Tree Depth on Model Accuracy

- The blue line shows clearly the overfitting of the model.
- The red line allows us to identify the optimal depth. (which is around 10).

Beyond this depth the model begins to memorize the training set (overfitting) instead of learning generalizable patterns.



# PRUNED TREE

**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Cost Complexity Pruning

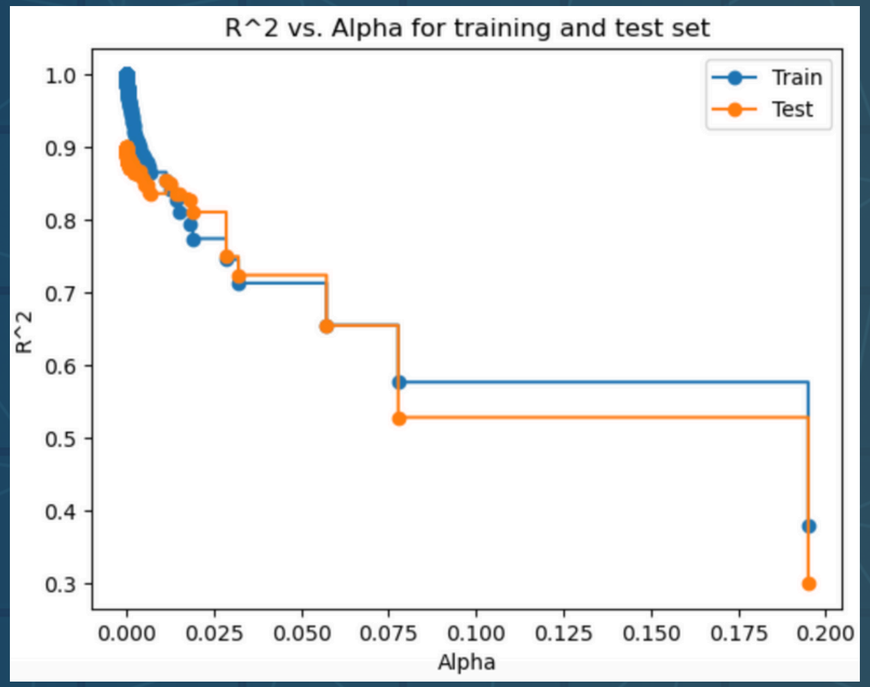
### Model Summary

```
Accuracy on training set: 0.992
Accuracy on test set: 0.899
Number of nodes:  317
Tree depth:  12
```

### Node Impurity

```
Average Impurity of the Tree nodes: 0.0548
Average Impurity of Leaf Nodes: 0.0065
```

### Impact of Tree Depth on Model Accuracy



# RANDOM FOREST

**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Random Forest

### Model Summary

```
Accuracy on training set: 0.988
Accuracy on test set: 0.940
```

```
Number of Trees of the forest: 5
Average Depth of the Trees of the forest: 16.4
```

### Node Impurity

```
Average Impurity of All Trees' Nodes: 0.0159
Average Impurity of Leaf Nodes Across Trees: -0.0000
```

| | Feature | Importance |
|---|---|---|
| 0 | Trip_Distance_km | 0.658028 |
| 7 | Per_Km_Rate | 0.193421 |
| 9 | Trip_Duration_Minutes | 0.079409 |
| 8 | Per_Minute_Rate | 0.054816 |
| 6 | Base_Fare | 0.006427 |
| 1 | Time_of_Day | 0.004069 |
| 3 | Passenger_Count | 0.001625 |
| 4 | Traffic_Conditions | 0.000933 |
| 5 | Weather | 0.000766 |
| 2 | Day_of_Week | 0.000506 |

# GRADIENT BOOSTING

**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Gradient Boosting

### DEFAULT PARAMETERS

- max_depth = 3
- n_estimators= 100
- learning_rate = 0.1

```
Accuracy on training set: 0.996
Accuracy on test set: 0.969
```

```
Average Impurity of All Nodes: 0.0605
Average Impurity of Leaf Nodes: 0.0356
```

### FIXED LEARNING RATE

- max_depth = 3
- n_estimators= 100
- learning_rate = 0.6

```
Accuracy on training set (learning rate = 0.6): 0.999
Accuracy on test set (learning rate = 0.6): 0.962
```

```
Average Impurity of All Nodes: 0.0158
Average Impurity of Leaf Nodes: 0.0096
```

# GRADIENT BOOSTING

**While Linear Regression and SVM prefer OneHotEncoding, for the training of decision trees we used LabelEncoding** (also because the categories had an order).

## Gradient Boosting

**FIXED NUMBER OF TREES**

- max_depth = 3
- n_estimators = 200
- learning_rate = 0.1

```
Accuracy on training set: 0.998
Accuracy on test set: 0.972
```

with respect to the default value (n_estimators = 100), increasing the number of trees doesn't improve so much the accuracy on the test set

```
Average Impurity of All Nodes: 0.0319
Average Impurity of Leaf Nodes: 0.0189
```

# NON- LINEAR MODELS

Ranking of best non-linear models

1. **Support Vector Regression with CV** (non-linear kernel) — 99,46%
2. **Gradient Boosting Regressor** (default parameters) — 96,9%
3. **Random forest** — 94%
4. **Pruned tree** (depth = 10, 7 features) — 89,9%
5. **Decisiont trees (depth =4)** — 83,3%

# CONCLUSION

## Outliers

Outliers didn't affect significantly the results. In particular, non-linear models are almost insesitive to outliers. The exclusion of outliers improved slightly the linear models (from 89% to 92%)

## Nonlinear models handle categorical variables better

In linear models, categorical variables were often not significant unless interaction terms were introduced between them.
Nonlinear models, on the other hand, can automatically capture these relationships without the need to manually create new terms.

## Trade-off between model complexity and model interpretability

Compared with linear regression, nonlinear models (such as Decision Trees, Random Forests, or SVMs with nonlinear kernels) performed better, however, they are more difficult to interpret.

# CHALLENGES

In class we primarily studied Support Vector Machines and Decision Trees for classification tasks.

For this group project, our goal was to make predictions.
We were able to apply SV and Decision Trees for regression tasks, but their interpretation and visualization was challenging.

# THANK YOU!