Predicting the transport cost of individual shipments

<u>Short description:</u>
Customers of the logistic industry would like to receive an immediate cost estimation for their transport orders; however, due to a manual/tedious process the dispatchers are unable to provide such. This mini project aims to accurately **predict the cost of a shipment** based on its attributes.

Dispatchers in the logistic industry must provide a cost immediately to the customer shipment orders

As-is situation

- **Logistic companies** usually simply act as the intermediary between a **customer** who wants to ship its goods and a **carrier** who can transport those shipments.
- Logistics **dispatchers** get customers orders and must assign a **cost** for the transportation of those shipments.

Issue

- The **process** to assign the cost is not straightforward, relies on human interactions/decisions and negotiations with the carrier can take several days.
- The customer wants to know a cost price **immediately**.
- The **dispatcher** has no way to provide an immediate cost approximation: he first needs to contact the potential carriers.

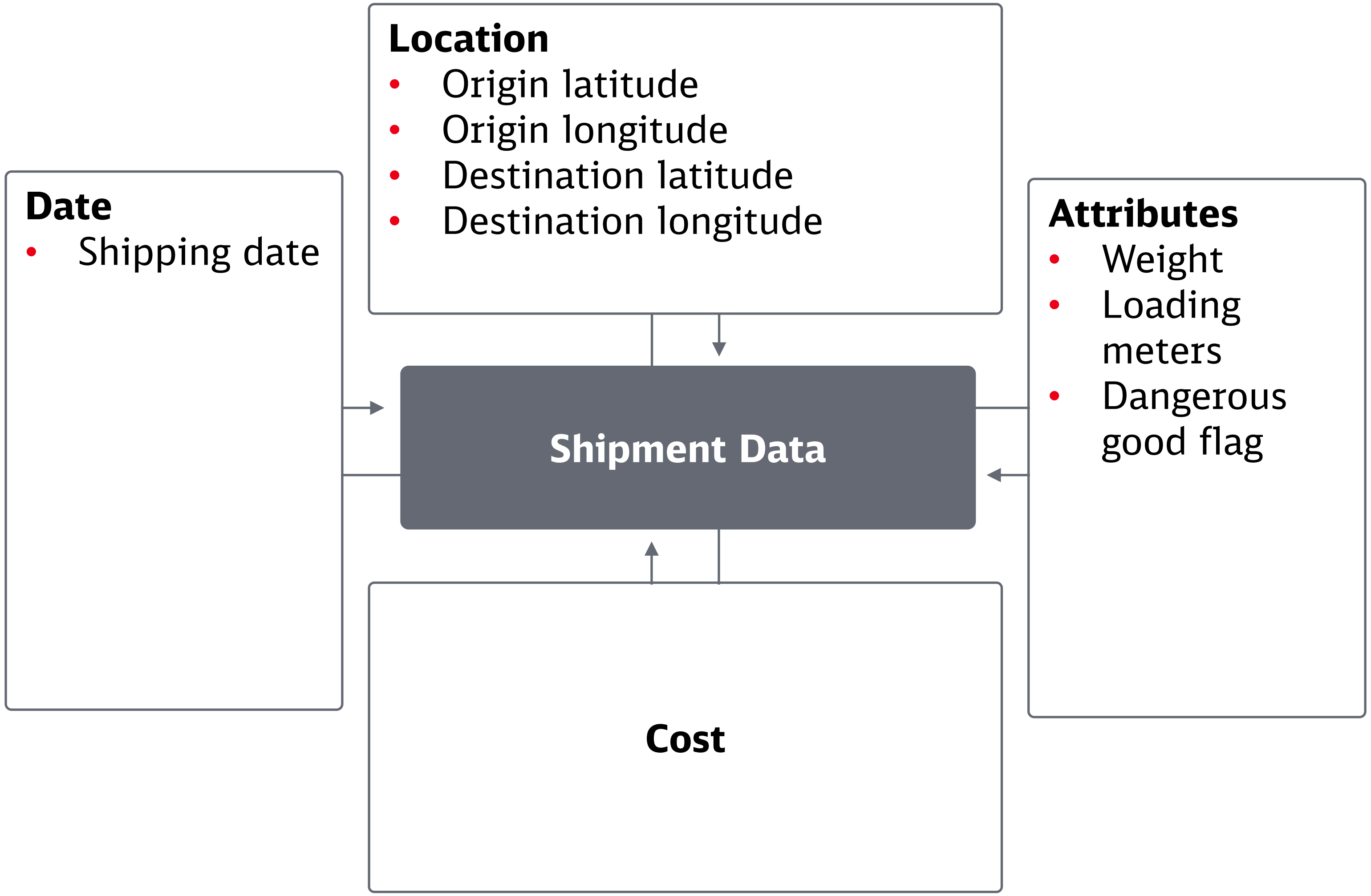
Objective

- Build a model which is able to **predict the cost** based on general shipment attributes
- **Evaluate** the performance of your model

A shipment data set was provided which includes locations, attributes, dates and the cost paid

Shipment Data

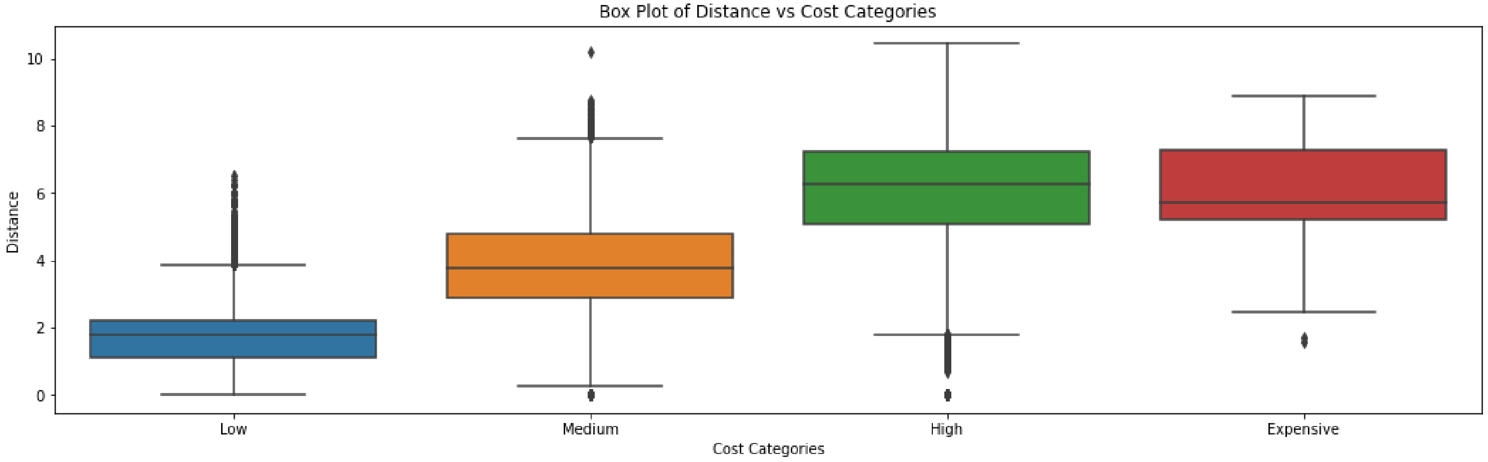
- **Historic shipment data** was provided by a logistic company, which will be kept as confidential
- Data has **9 columns** and over **250K rows**
- All shipment attribute values have previously being **normalized**
- Latitude, longitudes and dates have been **shifted**
- Column **cost** represents our **target value**

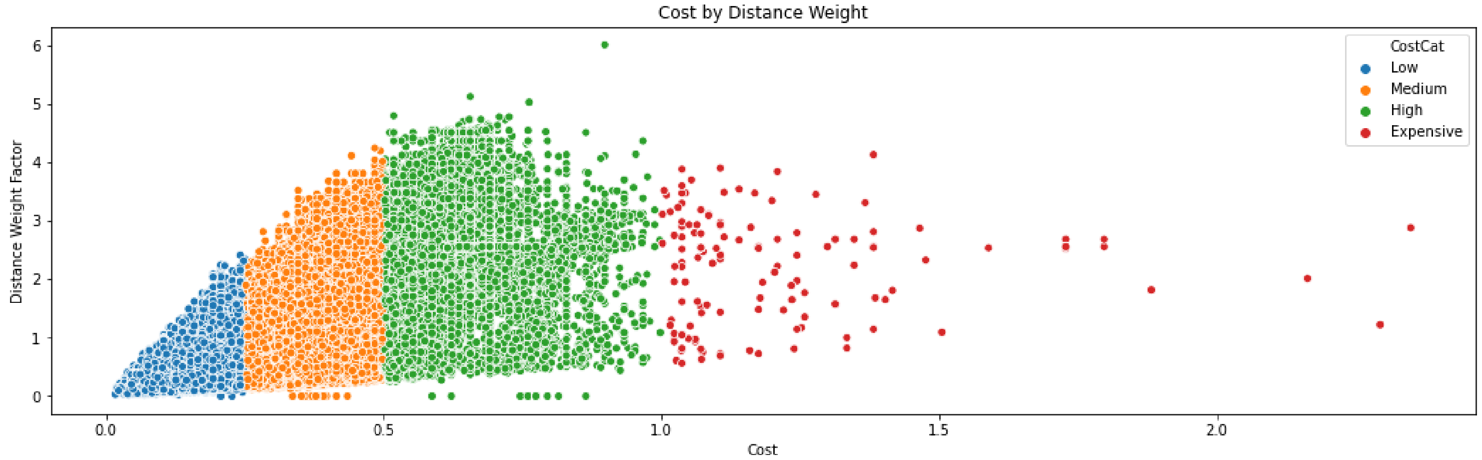


During EDA a simple Euclidean distance was computed, and other categorical columns were created

EDA & Categorization

- Simple **Euclidean distance** was computed from the origin and destination latitude and longitudes
- Weight by itself is not able to predict cost, however the **interaction between weight and distance** is a good predictor
- **Loading meters** were categorized into 5 types
- **Year and month** were extracted from the shipping date
- **Dangerous good flag** was dropped since it didn´t added any value





Five regression models from Scikit Learn were used to predict the cost of a shipment

Scikit Learn Models

- Simple Linear Regression
- Ridge Regression
- Lasso Regression
- Decision Tree Regressor
- Random Forest Regressor

Linear Regression

```
# Create and fit linear model
param_grid = {'fit_intercept': [True, False]}
lm = GridSearchCV(LinearRegression(), param_grid, cv=kfolds)
lm.fit(X_train, y_train)

print('Simple Linear Regression Trainning')
print('Best Parameters
print('Best Score
```

```
Simple Linear Regressi
Best Parameters : {'fi
Best Score      : 0.64
```

Decision Tree

```
dt = DecisionTreeRegressor()
dt_cv = cross_validate(dt, X_train, y_train, cv=kfolds)
dt.fit(X_train, y_train)

print('Decision Tree Trainning')
print('Mean Score       :', dt_cv['test_score'].mean())
```

```
Decision Tree Tr
DecisionTreeRegr
Mean Score
```

Random Forest

```
rf = RandomForestRegressor(criterion='mse', n_estimators=150)
rf_cv = cross_validate(rf, X_train, y_train, cv=kfolds)
rf.fit(X_train, y_train)

print('Random Forest Trainning')
print('Score       :', rf_cv['test_score'].mean())
```

```
Random Forest Trainning
Score       : 0.7887912398323781
```

The random forest model performed best, achieving an r^2 of 0.789 on test data

Results

- Models were evaluated on test data, using the r^2 from Scikit Learn
- First four models performed similarly, with values around 0.64
- Random forest model outperformed the rest, with a value of 0.789
- Random forest model should be used for future predictions

```
Score on Test Data
Linear Model Score      :   0.6441968929974466
Ridge Regression Score :   0.6441893968847746
Lasso Regression Score :   0.6441983495419472
Decision Tree Score     :   0.6647958139112322
Random Forest Score     :   0.7894916212306891
Random Forest will be used as Model!
```

Discussion

- The project results could be improved by the following actions
  - ➢ Calculate real driving kilometer distances instead of simple Euclidean
  - ➢ Identify from-to postal code combinations
  - ➢ Test random forest model approaches, such as gradient boosting and light GBM

Conclusion

- The logistic dispatcher employee must assign a cost to the transport of a shipment
- Five machine learning regression models were tested to predict the transport cost using shipment attributes as input
- The random forest model outperformed the rest, with an r^2 of 0.789
- The model performance could be improved by using real geo coordinates and testing more sophisticated models