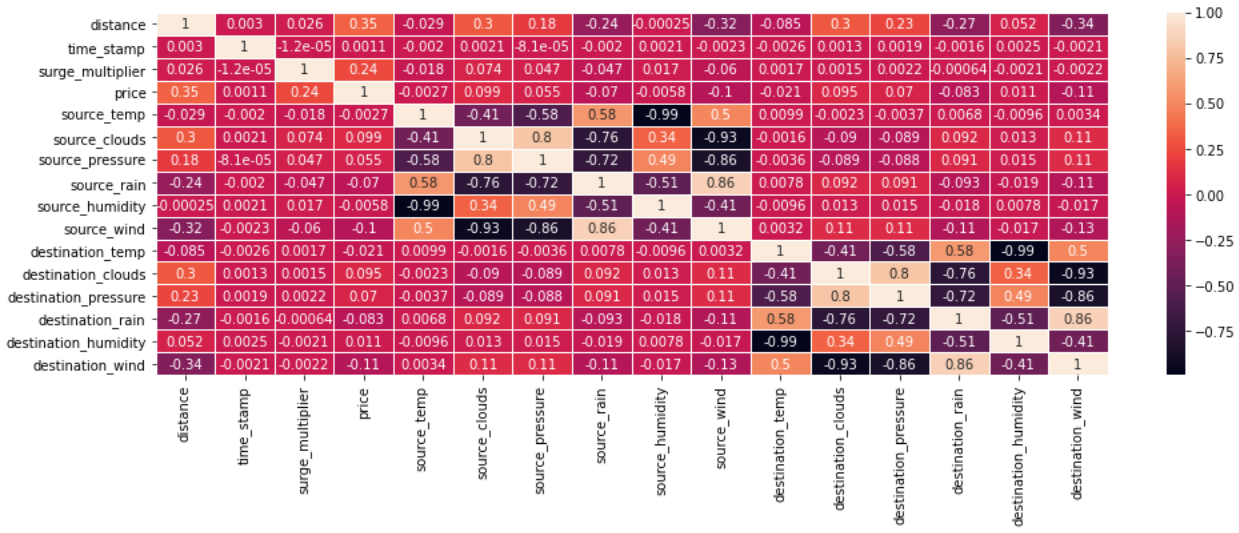
Data description

Since we are dealing with two different datasets we have had to find a connection between two of them, which we decided to connect them based on the source/destination locations
So we merged the weather dataset two times for the source and destination by grouping the data by location then rename the columns to their respected names (i.e : location -> destination location -> source)

preprocessing techniques

Remove dublications

We found out by analyzing the data that the two columns `product_id` and `name` `Are connected with each other which indicates that both are highly correlated so we opt to remove` `product_id` `since it's described as a duplication`



`from the figure we can see that there are multiple features that are considered highly correlated with each other that we can drop which are:`

- `source_humidity -> source_temp`
- `source_wind ->  source_cloud`
- `destination_humidity -> destination_temp`
- `destination_wind -> destination_cloud`

remove outliers

We used Zscore function to remove outliers from columns :

- 'distance',
- 'source_clouds',
- 'source_temp',
- 'source_pressure',
- 'source_rain',
- 'destination_clouds',
- 'destination_temp',
- 'destination_pressure',
- 'destination_rain'

Data Encoding

Using mean Encoding to encode data since it has great balance between efficiency and model complexity but the trade off was that we were forced to do the outlier removal before the encoding to make sure that the model will not overfitting

Feature selection

By studying the correlation between the data and the label we have found that price label doesn't have great correlation with the other so we choose based on correlation bigger than 0.1 Which lead us to use

- `Distance`
- `Source`
- `name`

Feature Scaling

We used MinMaxScaler() to scale the data

Model Training

First we split the data into 20% test and 80% train then we applied `two models with and without validation techniques:`

- linear_model.LinearRegression()

  ### Without cross validation

  `At degree = 12`

```
Train set size: 219064
Test set size: 54767
Train subset (MSE) for degree 12:  2.4452033145791487
Test subset (MSE) for degree 12:  2.4813097843306853
Training time: 7.6s
```

With cross validation cv = 9 and degree 12

model cross validation score is 2.458123420995886
model Test Mean Square Error 2.4813097843306853
Training time: 1m

- linear_model.Ridge()

Without cross validation

```
At degree = 12
Train set size: 219064
Test set size: 54767
Train subset (MSE) for degree 12:  2.72520925571384
Test subset (MSE) for degree 12:  2.7429110428911168
Training time: 7.3s
```

With cross validation cv = 9 and degree 12

model cross validation score is 2.7294260450383203
model Test Mean Square Error 2.7429110428911168
Training time: 1m : 28s

Conclusion

This dataset had proved a great challenge for us since it had so many label data which wasn't ordinal so it was challenging to encode the data
Also training the model was very consuming in terms of computation power needed so we had to figure out a way to reduce the dataset size without affecting the result which we had to use a combination between multiple data preprocessing techniques to achieve that