Untitled2

October 22, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from haversine import haversine, Unit
```

[2]:
```
df = pd.read_csv('uber.csv')
```

[3]:
```
df
```

[3]:
```
        Unnamed: 0                            key  fare_amount  \
0         24238194    2015-05-07 19:52:06.0000003          7.5
1         27835199    2009-07-17 20:04:56.0000002          7.7
2         44984355   2009-08-24 21:45:00.00000061         12.9
3         25894730    2009-06-26 08:22:21.0000001          5.3
4         17610152  2014-08-28 17:47:00.000000188         16.0
...            ...                            ...          ...
199995    42598914   2012-10-28 10:49:00.00000053          3.0
199996    16382965    2014-03-14 01:09:00.0000008          7.5
199997    27804658   2009-06-29 00:42:00.00000078         30.9
199998    20259894    2015-05-20 14:56:25.0000004         14.5
199999    11951496   2010-05-15 04:08:00.00000076         14.1

                pickup_datetime  pickup_longitude  pickup_latitude  \
0       2015-05-07 19:52:06 UTC        -73.999817        40.738354
1       2009-07-17 20:04:56 UTC        -73.994355        40.728225
2       2009-08-24 21:45:00 UTC        -74.005043        40.740770
3       2009-06-26 08:22:21 UTC        -73.976124        40.790844
4       2014-08-28 17:47:00 UTC        -73.925023        40.744085
...                         ...               ...              ...
199995  2012-10-28 10:49:00 UTC        -73.987042        40.739367
199996  2014-03-14 01:09:00 UTC        -73.984722        40.736837
199997  2009-06-29 00:42:00 UTC        -73.986017        40.756487
199998  2015-05-20 14:56:25 UTC        -73.997124        40.725452
199999  2010-05-15 04:08:00 UTC        -73.984395        40.720077

        dropoff_longitude  dropoff_latitude  passenger_count
0              -73.999512         40.723217                1
```

```
1              -73.994710         40.750325                1
2              -73.962565         40.772647                1
3              -73.965316         40.803349                3
4              -73.973082         40.761247                5
…                     …                 …              …
199995         -73.986525         40.740297                1
199996         -74.006672         40.739620                1
199997         -73.858957         40.692588                2
199998         -73.983215         40.695415                1
199999         -73.985508         40.768793                1

[200000 rows x 9 columns]
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   Unnamed: 0         200000 non-null  int64  
 1   key                200000 non-null  object 
 2   fare_amount        200000 non-null  float64
 3   pickup_datetime    200000 non-null  object 
 4   pickup_longitude   200000 non-null  float64
 5   pickup_latitude    200000 non-null  float64
 6   dropoff_longitude  199999 non-null  float64
 7   dropoff_latitude   199999 non-null  float64
 8   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB
```

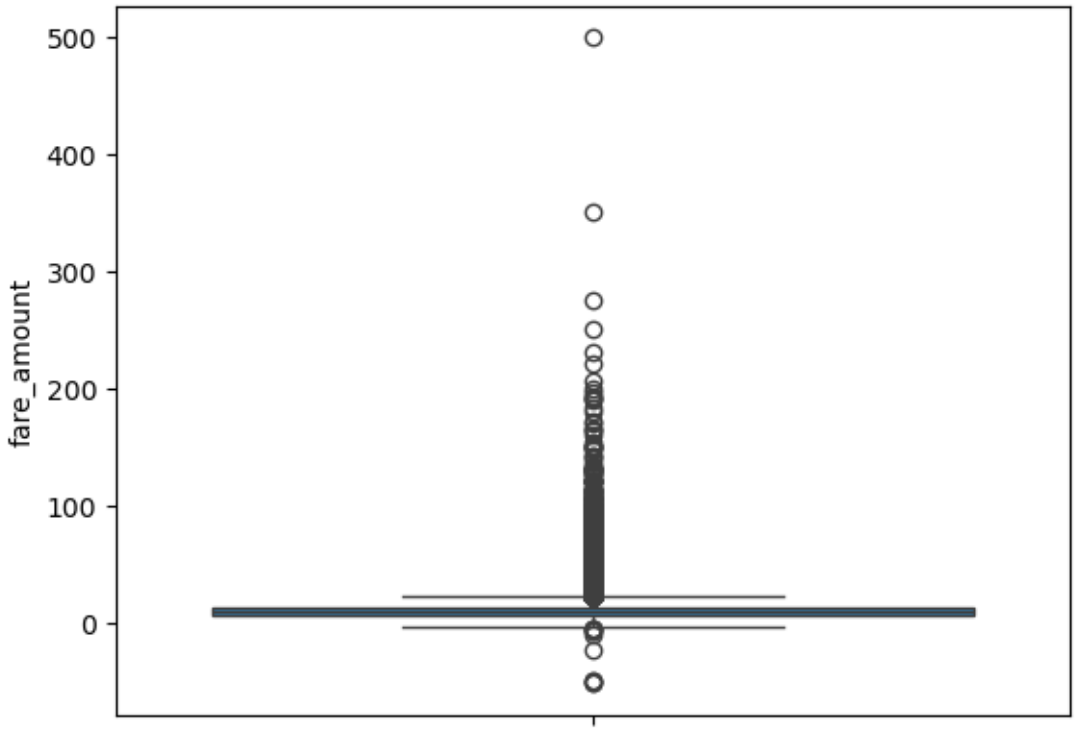
[5]:
```
df.isnull().sum()
```

[5]:
```
Unnamed: 0           0
key                  0
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

[6]:
```
df.shape
```

[6]:
```
(200000, 9)
```

[7]:
```
sns.boxplot(df['fare_amount'])
plt.show()
```



[8]:
```
from scipy.stats import zscore
df['fare_zscore'] = zscore(df['fare_amount'])
```

[9]:
```
df = df[(df['fare_zscore'] < 3) & (df['fare_zscore'] > -3)]
df.drop(columns='fare_zscore', inplace=True)
```

```
C:\Users\Chaitanya\AppData\Local\Temp\ipykernel_904\133405414.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.drop(columns='fare_zscore', inplace=True)
```

[10]:
```
df = pd.DataFrame(df)
```

[11]:
```
df
```

[11]:

	Unnamed: 0	key	fare_amount	\
0	24238194	2015-05-07 19:52:06.0000003	7.5	

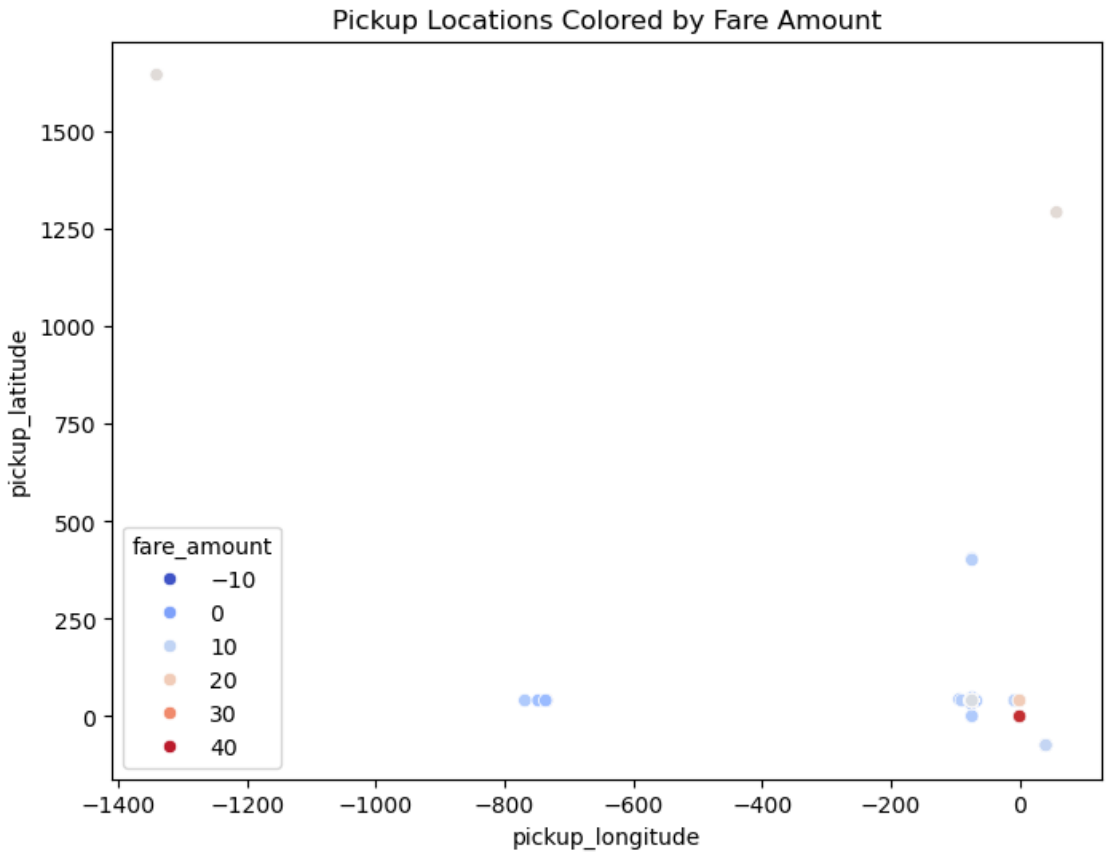
```
1          27835199    2009-07-17 20:04:56.0000002          7.7
2          44984355   2009-08-24 21:45:00.00000061         12.9
3          25894730    2009-06-26 08:22:21.0000001          5.3
4          17610152  2014-08-28 17:47:00.000000188         16.0
…               …                              …          …
199995     42598914   2012-10-28 10:49:00.00000053          3.0
199996     16382965    2014-03-14 01:09:00.0000008          7.5
199997     27804658   2009-06-29 00:42:00.00000078         30.9
199998     20259894    2015-05-20 14:56:25.0000004         14.5
199999     11951496   2010-05-15 04:08:00.00000076         14.1

                 pickup_datetime  pickup_longitude  pickup_latitude  \
0       2015-05-07 19:52:06 UTC        -73.999817        40.738354
1       2009-07-17 20:04:56 UTC        -73.994355        40.728225
2       2009-08-24 21:45:00 UTC        -74.005043        40.740770
3       2009-06-26 08:22:21 UTC        -73.976124        40.790844
4       2014-08-28 17:47:00 UTC        -73.925023        40.744085
…                           …               …                …
199995  2012-10-28 10:49:00 UTC        -73.987042        40.739367
199996  2014-03-14 01:09:00 UTC        -73.984722        40.736837
199997  2009-06-29 00:42:00 UTC        -73.986017        40.756487
199998  2015-05-20 14:56:25 UTC        -73.997124        40.725452
199999  2010-05-15 04:08:00 UTC        -73.984395        40.720077

        dropoff_longitude  dropoff_latitude  passenger_count
0               -73.999512         40.723217                1
1               -73.994710         40.750325                1
2               -73.962565         40.772647                1
3               -73.965316         40.803349                3
4               -73.973082         40.761247                5
…                      …                 …              …
199995          -73.986525         40.740297                1
199996          -74.006672         40.739620                1
199997          -73.858957         40.692588                2
199998          -73.983215         40.695415                1
199999          -73.985508         40.768793                1

[194550 rows x 9 columns]
```

[12]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='pickup_longitude', y='pickup_latitude', data=df,␣
 ↪hue='fare_amount', palette='coolwarm')
plt.title('Pickup Locations Colored by Fare Amount')
plt.show()
```



[13]:
```
df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'])
```

[14]:
```
df['day_of_week'] = df['pickup_datetime'].dt.dayofweek
df['hour_of_day'] = df['pickup_datetime'].dt.hour
```

[15]:
```
def haversine(lat1, lon1, lat2, lon2):
    R = 6371  # Radius of the Earth in kilometers
    dlat = np.radians(lat2 - lat1)
    dlon = np.radians(lon2 - lon1)
    a = np.sin(dlat / 2) ** 2 + np.cos(np.radians(lat1)) * np.cos(np.
 ↪radians(lat2)) * np.sin(dlon / 2) ** 2
    c = 2 * np.arctan2(np.sqrt(a), np.sqrt(1 - a))
    return R * c
```

[16]:
```
df['distance_km'] = df.apply(lambda row: haversine(row['pickup_latitude'],␣
 ↪row['pickup_longitude'],
                                                    row['dropoff_latitude'],␣
 ↪row['dropoff_longitude']), axis=1)
```

[34]:
```
df.dropna(inplace=True)
```

[35]:
```
df_imp = df[['passenger_count',␣
 ↪'day_of_week','hour_of_day','distance_km','fare_amount']]
```

[48]:
```
df_imp
```

[48]:

	passenger_count	day_of_week	hour_of_day	distance_km	fare_amount
0	1	3	19	1.683323	7.5
1	1	4	20	2.457590	7.7
2	1	0	21	5.036377	12.9
3	3	4	8	1.661683	5.3
4	5	3	17	4.475450	16.0
…	…	…	…	…	…
199995	1	6	10	0.112210	3.0
199996	1	4	1	1.875050	7.5
199997	2	0	0	12.850319	30.9
199998	1	2	14	3.539715	14.5
199999	1	5	4	5.417783	14.1

[194549 rows x 5 columns]

[49]:
```
from sklearn.preprocessing import MinMaxScaler
```

[50]:
```
scale = MinMaxScaler()
```

[53]:
```
df_imp['distance_km'] = scale.fit_transform(df[['distance_km']])
df['distance_km'] = scale.fit_transform(df[['distance_km']])
```

```
C:\Users\Chaitanya\AppData\Local\Temp\ipykernel_904\589624380.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_imp['distance_km'] = scale.fit_transform(df[['distance_km']])
```

[66]:
```
df_imp
```

[66]:

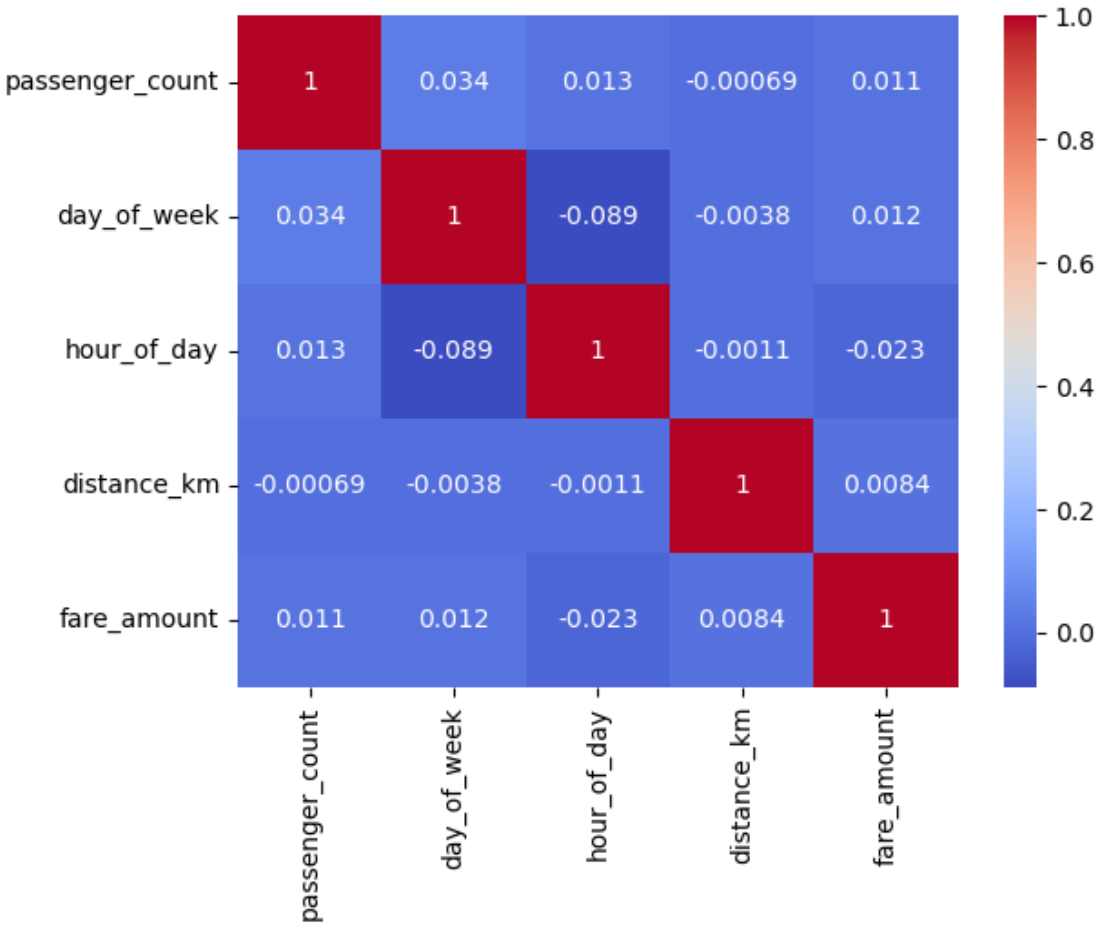
	passenger_count	day_of_week	hour_of_day	distance_km	fare_amount
0	1	3	19	0.000103	7.5
1	1	4	20	0.000150	7.7
2	1	0	21	0.000307	12.9
3	3	4	8	0.000101	5.3
4	5	3	17	0.000273	16.0
…	…	…	…	…	…
199995	1	6	10	0.000007	3.0
199996	1	4	1	0.000114	7.5
199997	2	0	0	0.000783	30.9

```
199998                1            2            14      0.000216          14.5
199999                1            5             4      0.000330          14.1

[194549 rows x 5 columns]
```

[54]:
```
X = df[['passenger_count', 'day_of_week','hour_of_day','distance_km']]
y = df['fare_amount']
```

[55]:
```
corr_matrix = df_imp.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.show()
```



[56]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error , r2_score
from sklearn.linear_model import LinearRegression
```

[57]:
```
X.dropna(inplace=True)
X.isnull().sum()
```

```
C:\Users\Chaitanya\AppData\Local\Temp\ipykernel_904\2352594663.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X.dropna(inplace=True)
```

[57]:
```
passenger_count    0
day_of_week        0
hour_of_day        0
distance_km        0
dtype: int64
```

[58]:
```
x_train , x_test , y_train , y_test = train_test_split(X,y,test_size =0.2,␣
↪random_state=42)
```

[59]:
```
rf_model = RandomForestRegressor(n_estimators=50, random_state=42)
lin_model = LinearRegression()
```

[60]:
```
rf_model.fit(x_train,y_train)
```

[60]:
```
RandomForestRegressor(n_estimators=50, random_state=42)
```

[61]:
```
lin_model.fit(x_train,y_train)
```

[61]:
```
LinearRegression()
```

[62]:
```
y_pred_rf = rf_model.predict(x_test)
y_pred_lin = lin_model.predict(x_test)
```

[63]:
```
r2 = r2_score(y_test, y_pred_rf)
print(f"R-squared (R2 Score): {r2}")
```

```
R-squared (R2 Score): 0.7155370197495386
```

[64]:
```
r2 = r2_score(y_test, y_pred_lin)
print(f"R-squared (R2 Score): {r2}")
```

```
R-squared (R2 Score): 0.0013684380851716194
```

[ ]:
```

```