Classification

Question(a)

Data Pre-Processing

First, we described the data to see low variance variables and whether there exist some outliers:

id	source_tracking_id	source_lng	source_lat	target_lng	target_lat	grid_distance	expected_use_time	urgency	hour
00	5.096040e+05	509604.000000	509604.000000	509604.000000	509604.000000	509604.000000	509604.000000	509604.000000	509604.000000
00	2.100076e+18	121.534923	39.179897	121.534882	39.179971	1078.274900	441.655107	1572.033695	14.482592
27	4.797965e+12	0.150718	0.113594	0.150752	0.113615	1124.569317	405.080785	4344.556228	3.310272
00	2.100070e+18	119.876654	36.064995	121.059274	38.826421	0.000000	1.000000	-340771.000000	6.000000
00	2.100070e+18	121.444174	39.117340	121.444254	39.117201	330.000000	189.000000	859.000000	12.000000
00	2.100080e+18	121.523930	39.161311	121.523587	39.161241	869.000000	354.000000	1752.000000	14.000000
00	2.100080e+18	121.591344	39.218011	121.591347	39.218921	1572.000000	584.000000	2590.000000	17.000000
00	2.100080e+18	122.260124	39.705013	122.260124	39.695211	429173.000000	9246.000000	11345.000000	23.000000

We found that `grid_distance` has some outliers because 75% quantile is 1572 whereas its max value is 429173, which is approximate 300 times of 75% quantile. Therefore, we removed those observations by counting the number of values which more than 10000 meters, 20000 meters and decided deleting observations whose `grid_distance` is more than 10000 meters.

```
# remove outliers
df_train = df_train.drop(df_train[df_train.grid_distance>10000].index)
```

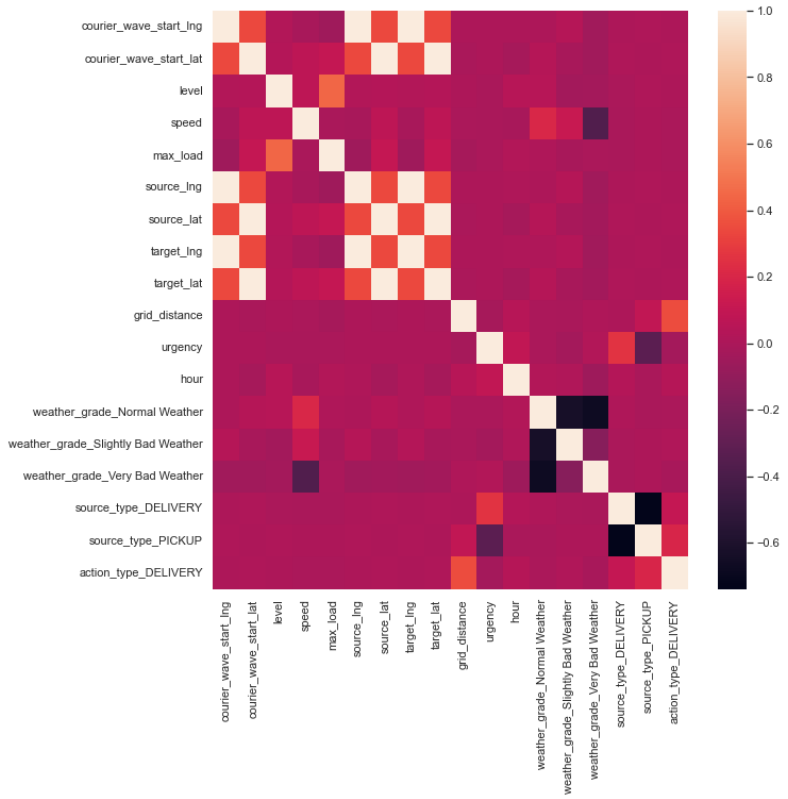
Second, we down-sampled the training data set into 10000 and dropped some unreasonable variables like id and some variables we cannot use to predict the testing data set. Then, we converted some discrete variables into numerical data by getting dummies. Incidentally, we generated our labels which represents action type is pick-up when label equals 1.

```
# down-sample randomly
df = df_train.sample(n=10000,random_state=666,axis=0)

# drop unreasonable variables and variables we can't use in test set
df = df.drop(columns = ['courier_id', 'wave_index', 'tracking_id','date','group',
                        'id','shop_id','aoi_id','source_tracking_id','expected_use_time'])

# convert into numerical data
df = pd.get_dummies(df, drop_first=True)
df['action_type_DELIVERY'] = np.array(df['action_type_PICKUP']==0).astype(int)
df = df.drop(columns=['action_type_PICKUP'])
```

Third, we standardized the data sample and ran a heat map to reveal the correlations between all the variables to see whether independent variables have strong correlations and which independent variables have correlations with label.



According to the figure above, the following sets of variables have strong correlation:

```
(courier_wave_start_lng,source_lng,target_lng), (courier_w
ave_start_lat, source_lat, target_lat)
```

Additionally, the following variables are related to action_type_PICKUP more than 1%:

```
(urgency, grid_distance, source_type_PICKUP, source_type_D
ELIVERY, hour, level)
```

Therefore, we chose them tentatively as our independent variables in our baseline model and in exploring models.

Question(b)

Baseline Model

In [9]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression as LogitReg

LR = LogitReg(max_iter=10000)
param_grid = [{'C':np.arange(0.1,1,0.1)}]
grid = GridSearchCV(LR, param_grid,cv=5)
grid.fit(X_train,y_train)
print("The best alpha parameter is",grid.best_params_)
print('\n')
print("The corresponding R-squared is %0.4f." % grid.best_score_)
```

```
The best alpha parameter is {'C': 0.9}


The corresponding R-squared is 0.7281.
```

In [10]:
```
from sklearn.metrics import f1_score
from sklearn.metrics import mean_absolute_error
LR = LogitReg(max_iter=10000,C=0.9).fit(X_train, y_train)
y_pred = LR.predict(X_test)
print('F1 score of classification baseline model is:',f1_score(y_test, y_pred),
      '\nMAE of classification baseline model is:',mean_absolute_error(y_test, y_pred))
```

```
F1 score of classification baseline model is: 0.7253989361702129
MAE of classification baseline model is: 0.2753333333333333
```

We used GridSearchCV to cross validate the logistic regression model which is our baseline model. We found that when C parameter equals 0.9, the corresponding R-squared is the highest, 0.7281. Then, we refitted the model with C=0.9 and got the F1 score of 0.725. This is not bad but still not good enough. As a result, we need to try more models and feature engineering.

Question(c)

More Complex Models

K-NN:

```
from sklearn.neighbors import KNeighborsClassifier
# We use the mean and standard deviation of the training set to standardize both the training and testing sets.
scaler = StandardScaler().fit(X_train)

X_train_st = scaler.transform(X_train)

X_test_st = scaler.transform(X_test)
```

```
knn_clf = KNeighborsClassifier(n_neighbors = 5).fit(X_train_st,y_train)
pred_ret = knn_clf.predict(X_test_st)
```

```
print('F1 score of K-NN model is:',f1_score(y_test, pred_ret),
      '\nMAE of K-NN model is:',mean_absolute_error(y_test, pred_ret))
```

```
F1 score of K-NN model is: 0.7580174927113703
MAE of K-NN model is: 0.249
```

Random Forest:

```
: from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier(n_estimators=100)
RF.fit(X_train, y_train)
RF_pred = RF.predict(X_test)
```

```
: print('F1 score of Random Forest model is:',f1_score(y_test, RF_pred ),
      '\nMAE of Random Forest model is:',mean_absolute_error(y_test, RF_pred ))
```

```
F1 score of Random Forest model is: 0.7657920310981535
MAE of Random Forest model is: 0.241
```

XGBT:

```
import xgboost as xgb
xgb_reg = xgb.XGBClassifier(objective ='binary:logistic', eval_metric='logloss', colsample_bynode=0.8,use_label_encoder=False,
                        learning_rate = 0.1, gamma = 0.001,max_depth = 5,  n_estimators = 50)

xgb_reg.fit(X_train,y_train)
preds = xgb_reg.predict(X_test)
```

```
print('F1 score of XGBT model is:',f1_score(y_test, preds),
     '\nMAE of XGBT model is:',mean_absolute_error(y_test, preds))
```

```
F1 score of XGBT model is: 0.7988826815642459
MAE of XGBT model is: 0.216
```

After trying different models, the F1 score improved significantly. Among these models, XGBT has a F1 score reaching 0.8. As a result, we chose XGBT as our final model in feature engineering and predicting our test data set.

Question(d)

Feature Engineering

First, we polynomial features and found degree=1 is the best.

```
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import Ridge

poly_pipe = make_pipeline(PolynomialFeatures(degree=1), xgb.XGBClassifier(objective ='binary:logistic', eval_metric='logloss', colsample_bynod
                        learning_rate = 0.1, gamma = 0.001,max_depth = 5,  n_estimators = 50))
poly_pipe.fit(X_train, y_train)

print("The out-of-sample R-squared with polynomial features and ridge regression is %0.4f."%poly_pipe.score(X_test, y_test))
```

```
The out-of-sample R-squared with polynomial features and ridge regression is 0.7823.
```

degree = 1,2,3 have very close R-squared, and degree = 1 is the highest, so we choose degree=1.

Second, we clustered the geographic information into 3 categories and added them as dummy variables into our model.

```
# combine lat and lng into location
courier_location = list(map(lambda x,y: [x,y], courier_lng, courier_lat))
target_location = list(map(lambda x,y: [x,y], target_lng, target_lat))
source_location = list(map(lambda x,y: [x,y], source_lng, source_lat))
```

```
# clustering locations
from sklearn.cluster import AgglomerativeClustering as HieClustering

clustering1 = HieClustering(n_clusters=3,linkage='ward').fit(courier_location)
courier_cluster = clustering1.labels_

clustering2 = HieClustering(n_clusters=3,linkage='ward').fit(target_location)
target_cluster = clustering2.labels_

clustering3 = HieClustering(n_clusters=3,linkage='ward').fit(source_location)
source_cluster = clustering3.labels_

# add and refit the RandomForest model
df['courier_geo'] = courier_cluster
df['target_geo'] = target_cluster
df['source_geo'] = source_cluster
```

Actually, this did not improve our F1 score too much, so we conducted RFECV to select independent variables.

Third, we conducted feature selection by RFECV using all the variables except id.

```
X_train_scaled = df_standard.drop(columns=['action_type_DELIVERY'])

# Use xgbc as the model for RFE and CV.

rfe = RFECV(xgb.XGBClassifier(objective ='binary:logistic', eval_metric='logloss', colsample_bynode=0.8,use_label_encoder=False,
                          learning_rate = 0.1, gamma = 0.001,max_depth = 15,  n_estimators = 100), cv=10)

rfe.fit(X_train_scaled, y)

print("The the support of the selected features are: ",rfe.support_)

print('\n')

print("The selected features are: ",df_standard.drop(columns=['action_type_DELIVERY']).columns[rfe.support_])
```

```
The the support of the selected features are:  [ True  True False False False False  True  True  True  True  True False
 False  True  True  True False False False False False False False False
 False False False False False False False False False False False False]


The selected features are:  Index(['courier_wave_start_lng', 'courier_wave_start_lat', 'source_lat',
       'target_lng', 'target_lat', 'grid_distance', 'urgency',
       'weather_grade_Slightly Bad Weather', 'weather_grade_Very Bad Weather',
       'source_type_DELIVERY', 'source_type_PICKUP'],
      dtype='object')
```

Fourth, we split the whole training data set into train set and test set and refitted the model using the recommended features.

```
xgb_reg = xgb.XGBClassifier(objective ='binary:logistic', eval_metric='logloss', colsample_bynode=0.8,use_label_encoder=False,
                          learning_rate = 0.1, gamma = 0.001,max_depth = 15,  n_estimators = 500)

xgb_reg.fit(X_train,y_train)
preds = xgb_reg.predict(X_test)

print('F1 score of XGBT model is:',f1_score(y_test, preds),
      '\nMAE of XGBT model is:',mean_absolute_error(y_test, preds))
```

```
F1 score of XGBT model is: 0.9627017372950155
MAE of XGBT model is: 0.037565410779696495
```

Finally, the F1 score improve to 0.96.

Question(e)

We used the whole training set to fit our model and made the parameters of XGBT better and predicted our testing set. We found a mistake in test data set because the tracking id and source_type were swapped.

According to our model, source_type, grid_distance and urgency are the most important features in predicting the courier’s action type. Weather condition and geographic information can help predict courier’s next move. But level, max load of courier, speed, hour seemed no effects on courier’s action type.