



Anticiper le retard de vol des avions

PROJET 4

SANDRINE FRANIATTE

BNP PARIBAS DDAA -
OPENCLASSROOMS

Constat

En Europe, les retards d'avion ont doublé en 2018.

Le trafic aérien en Europe a connu une hausse de 3,8% entre 2017 et 2018 mais les retards ont quant à eux augmenté en flèche de 105%.

L'inefficacité de l'espace aérien européen a coûté 17,6 milliards d'euros à l'Union européenne" en 2018 et "334 millions de passagers ont été affectés par les perturbations"

Intérêt du sujet

Aéroports et Compagnies aériennes:

- Est-il possible de prédire un retard?

Particuliers:

- Quelle est la probabilité que mon vol soit en retard?

Source de données

Les données ont été collectées et publiées par le Département US des Transports et sont disponibles à l'adresse suivante: <https://www.transtats.bts.gov/>

L'année 2016 aux USA



5635978

VOLS

12

COMPAGNIES AERIENNES

309

AEROPORTS

Exploration des données

Données sélectionnées – 01/2016 -> 06/2016

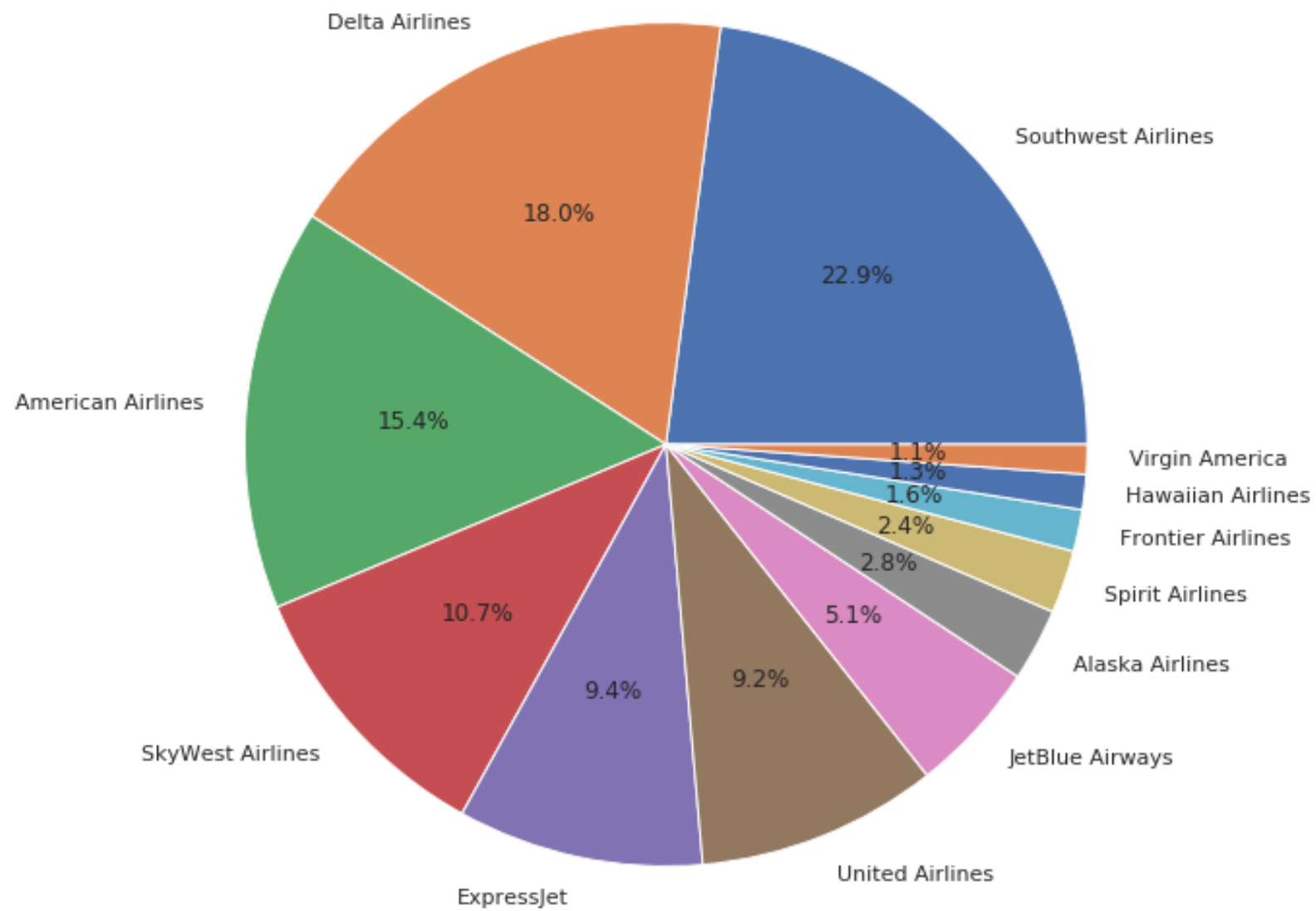
	DAY_OF_MONTH	MONTH	UNIQUE_CARRIER	ORIGIN	DEST	CRS_DEP_TIME	CRS_ARR_TIME	DEP_DELAY	ARR_DELAY	CANCELLED	DIVERTED	CARRIER
0	6	1	AA	DFW	DTW	1100.0	1438.0	-3.0	-6.0	0.0	0.0	
1	7	1	AA	DFW	DTW	1100.0	1438.0	-4.0	-12.0	0.0	0.0	
2	8	1	AA	DFW	DTW	1100.0	1438.0	-5.0	7.0	0.0	0.0	
3	9	1	AA	DFW	DTW	1100.0	1438.0	2.0	-5.0	0.0	0.0	
4	10	1	AA	DFW	DTW	1100.0	1438.0	100.0	113.0	0.0	0.0	

Colonnes sélectionnées

- DAY_OF_MONTH: date of the flight
- MONTH: month of the flight
- UNIQUE_CARRIER: Airline carrier
- ORIGIN: Departure Airport
- DEST: Destination
- CRS_DEP_TIME: Planned Departure time
- CRS_ARR_TIME: Planned Arrival time
- DEP_DELAY: Departure delay, in minutes
- ARR_DELAY: Arrival delay, in minutes
- CANCELLED: was the flight cancelled
- DIVERTED: was the flight diverted
- CARRIER_DELAY: in minutes
- WEATHER_DELAY: in minutes
- NAS_DELAY: in minutes
- SECURITY_DELAY: in minutes
- LATE_AIRCRAFT_DELAY: in minutes

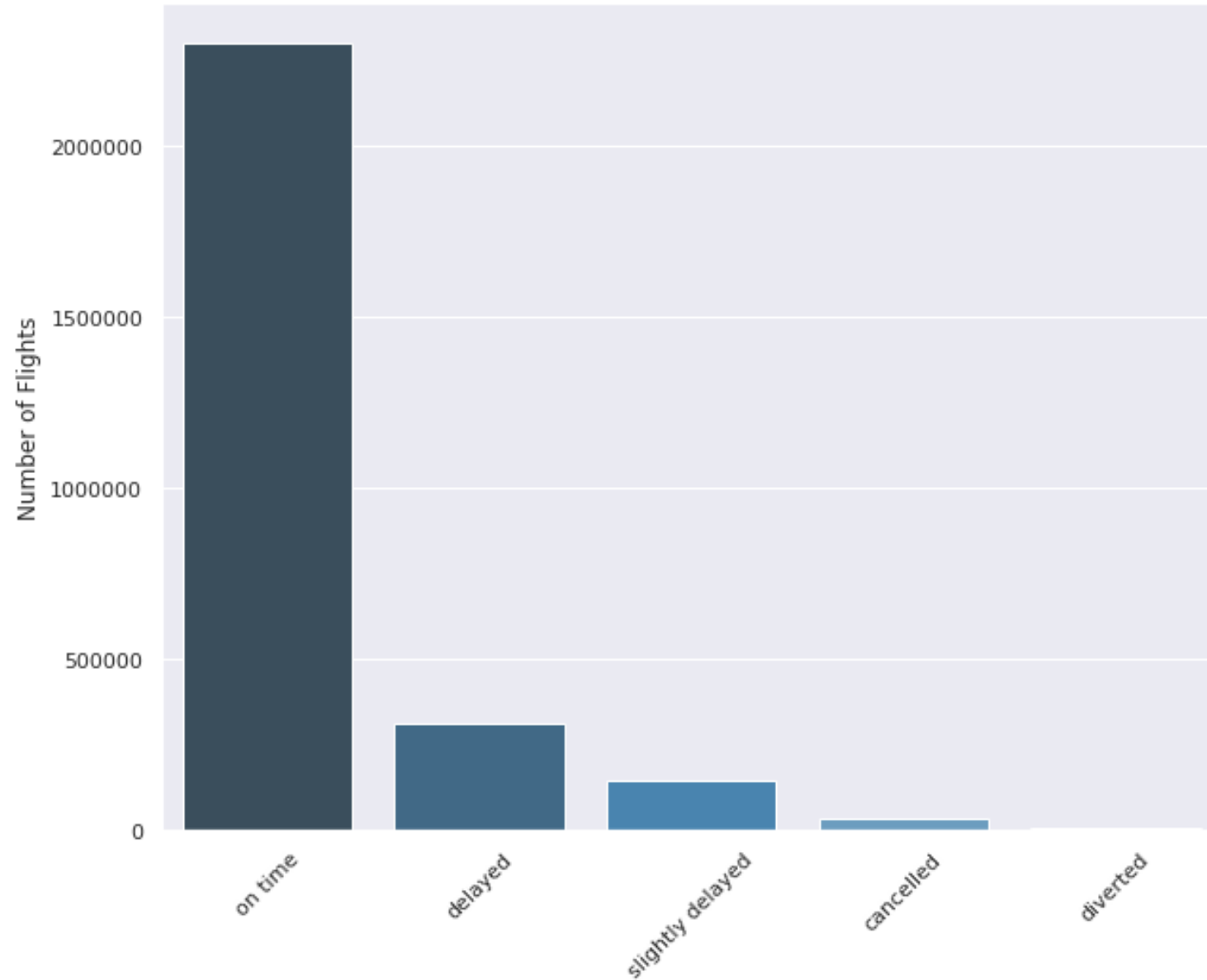
Les compagnies

% of flights per company



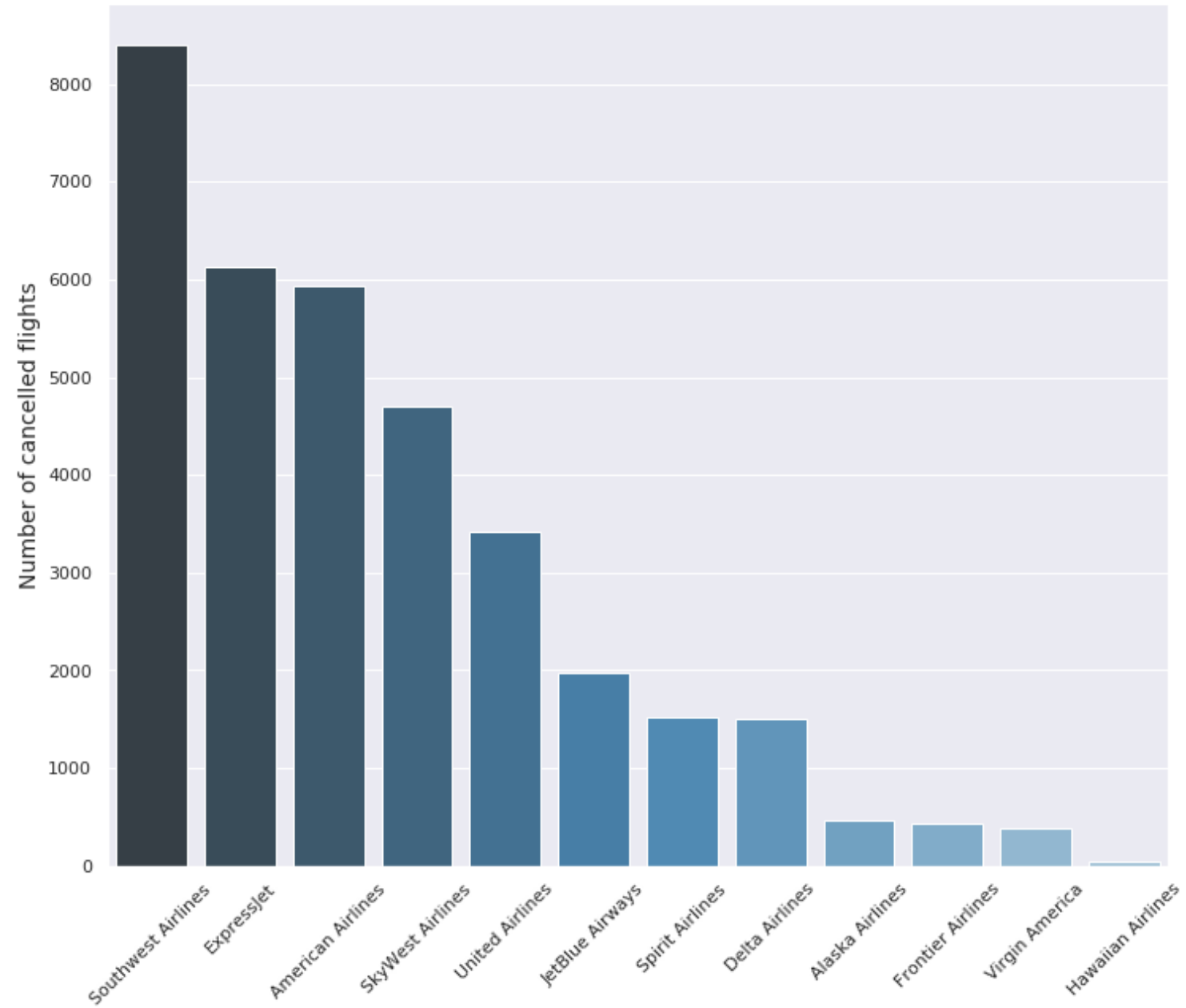
Le statut des vols

Number of Flights per status



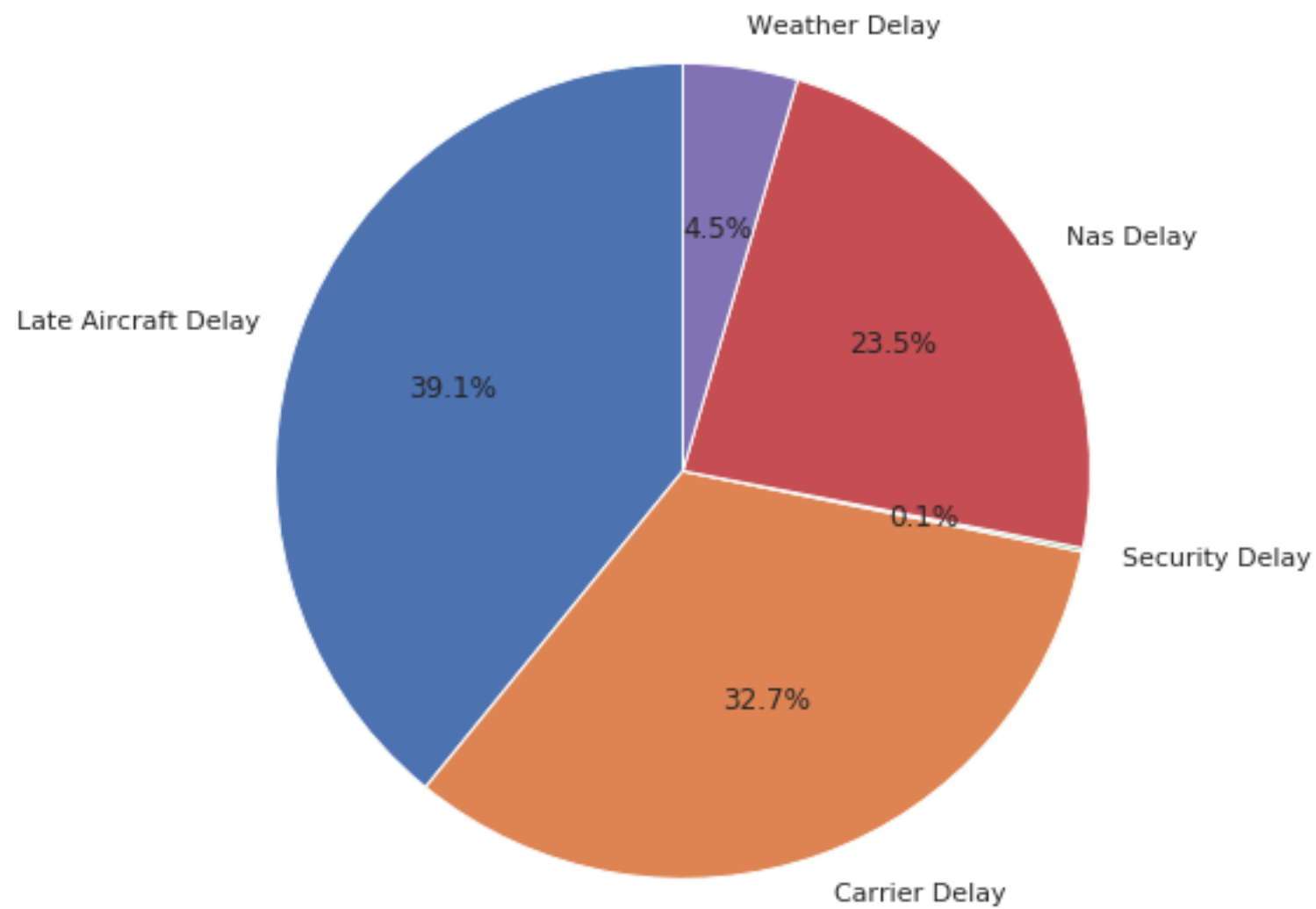
Les annulations

Cancelled flights by airlines



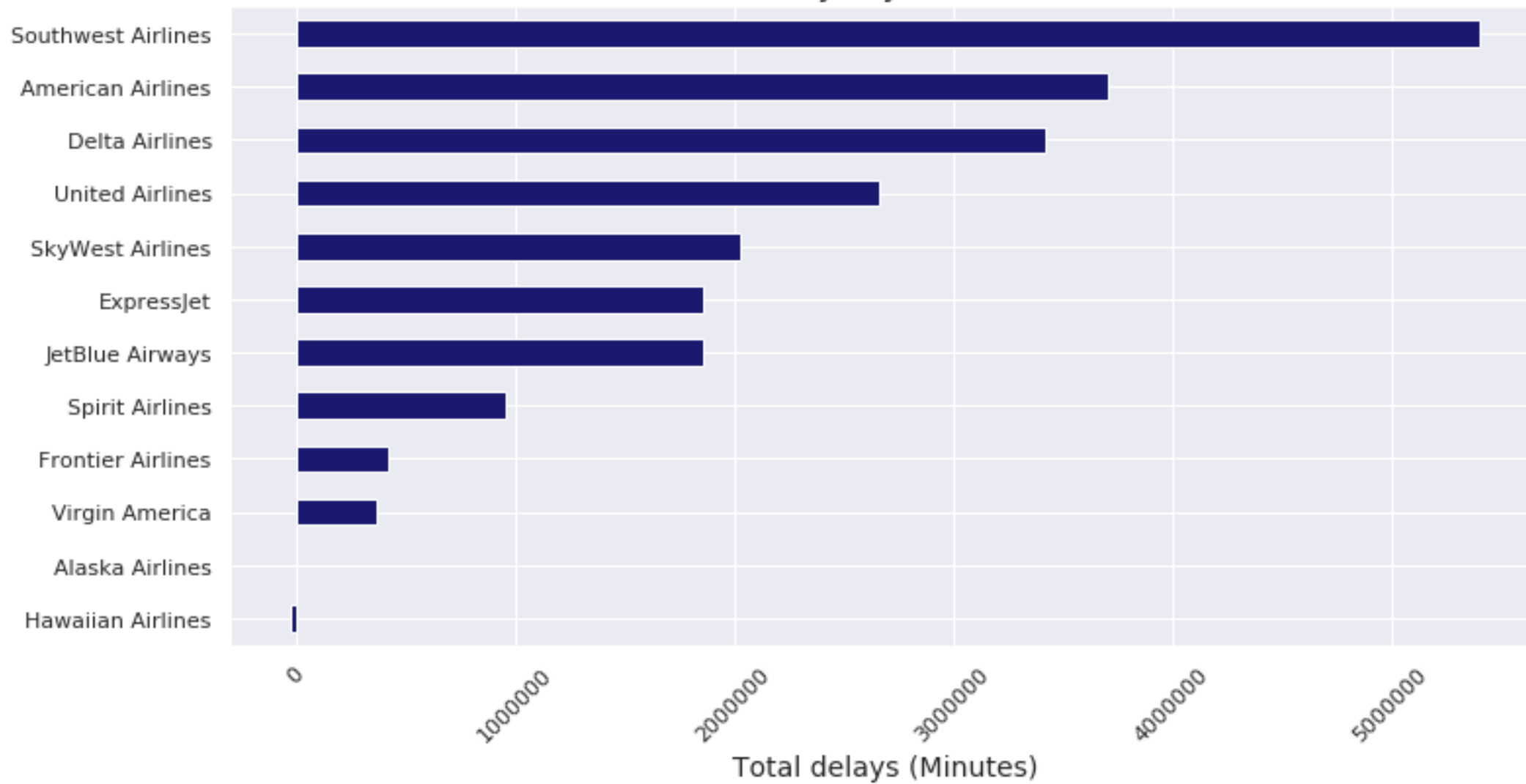
Les retards - causes

US Delay Cause Impact



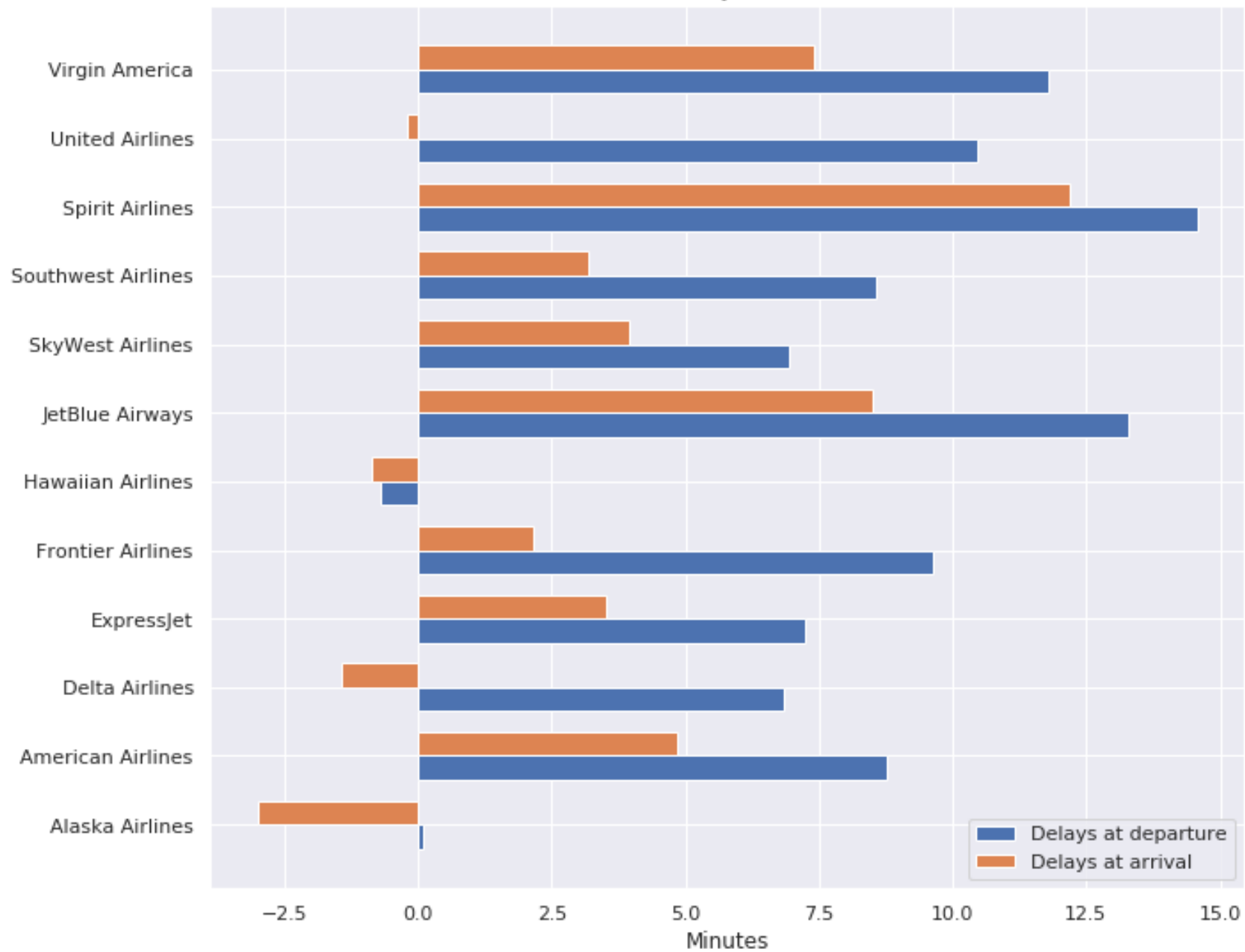
Les retards – Comparaison par compagnies

Delays by airlines

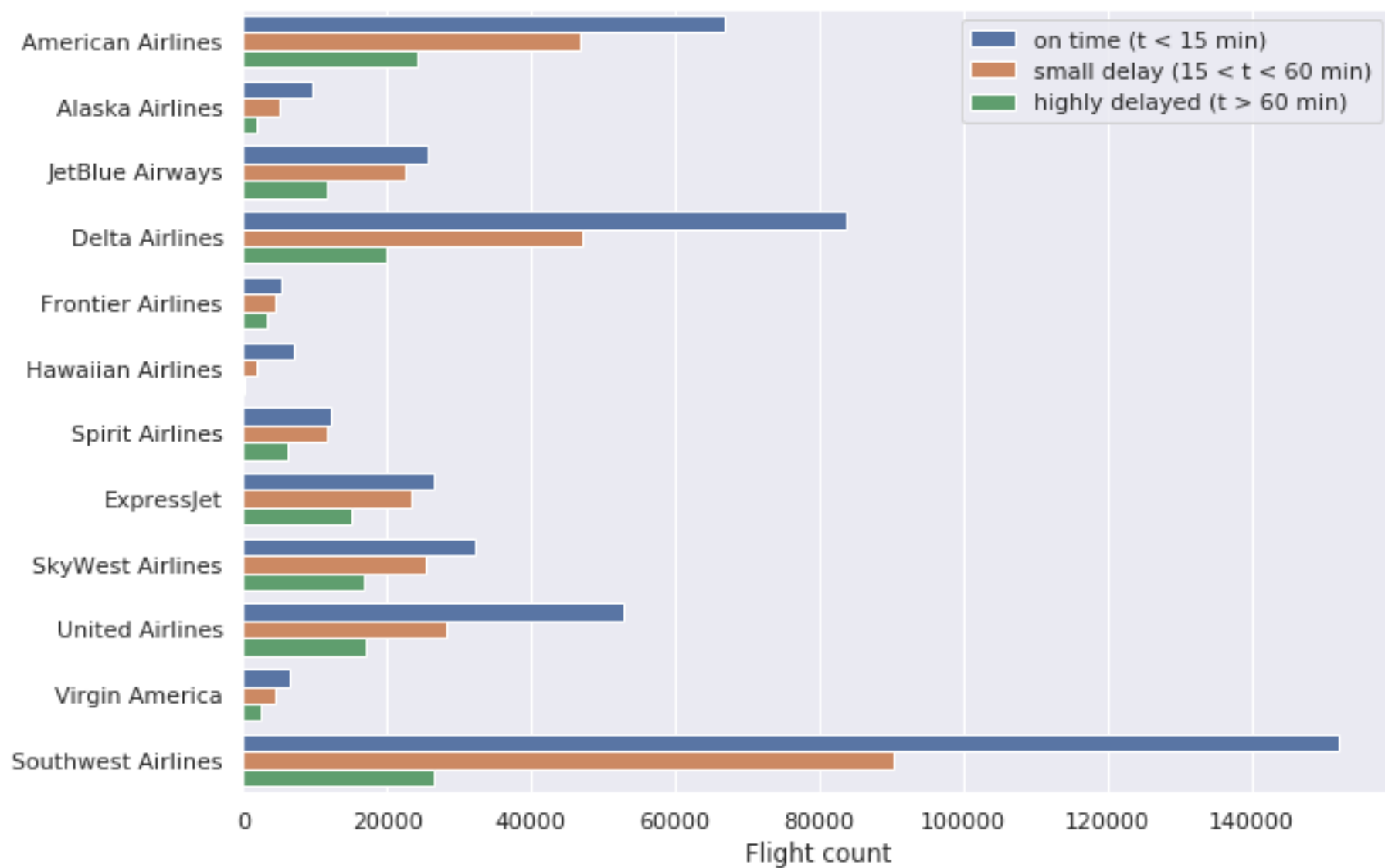


Les retards – au
départ/à l'arrivée

Mean delay (minutes)

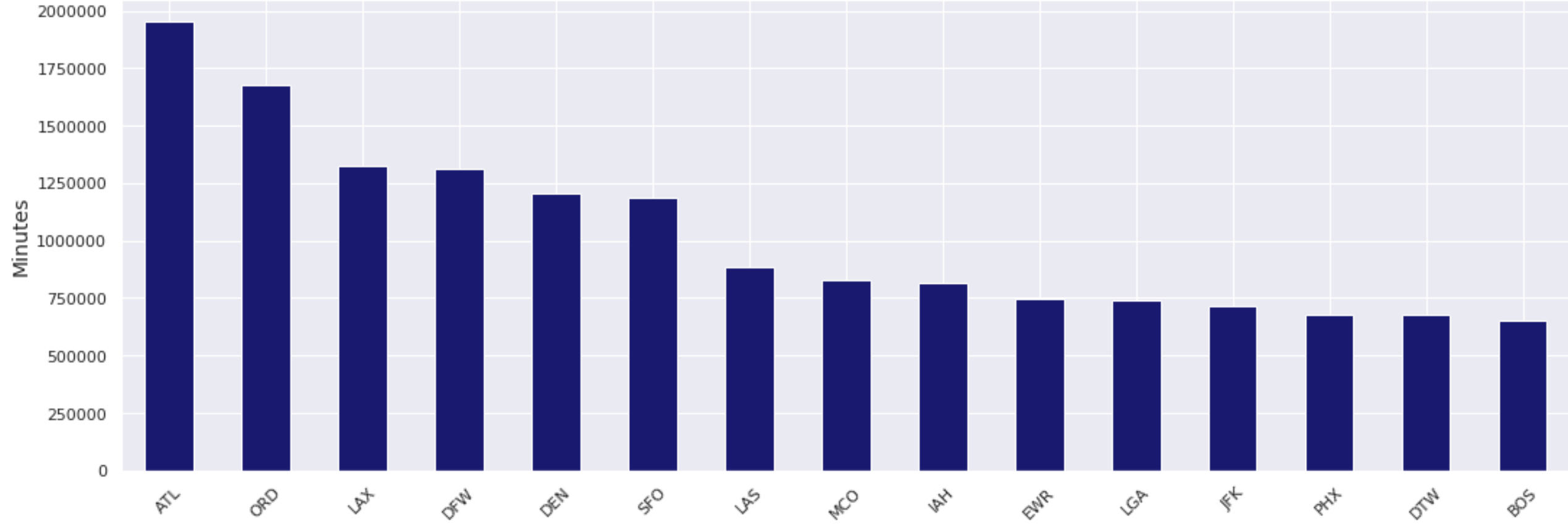


Les retards – Importance par compagnie

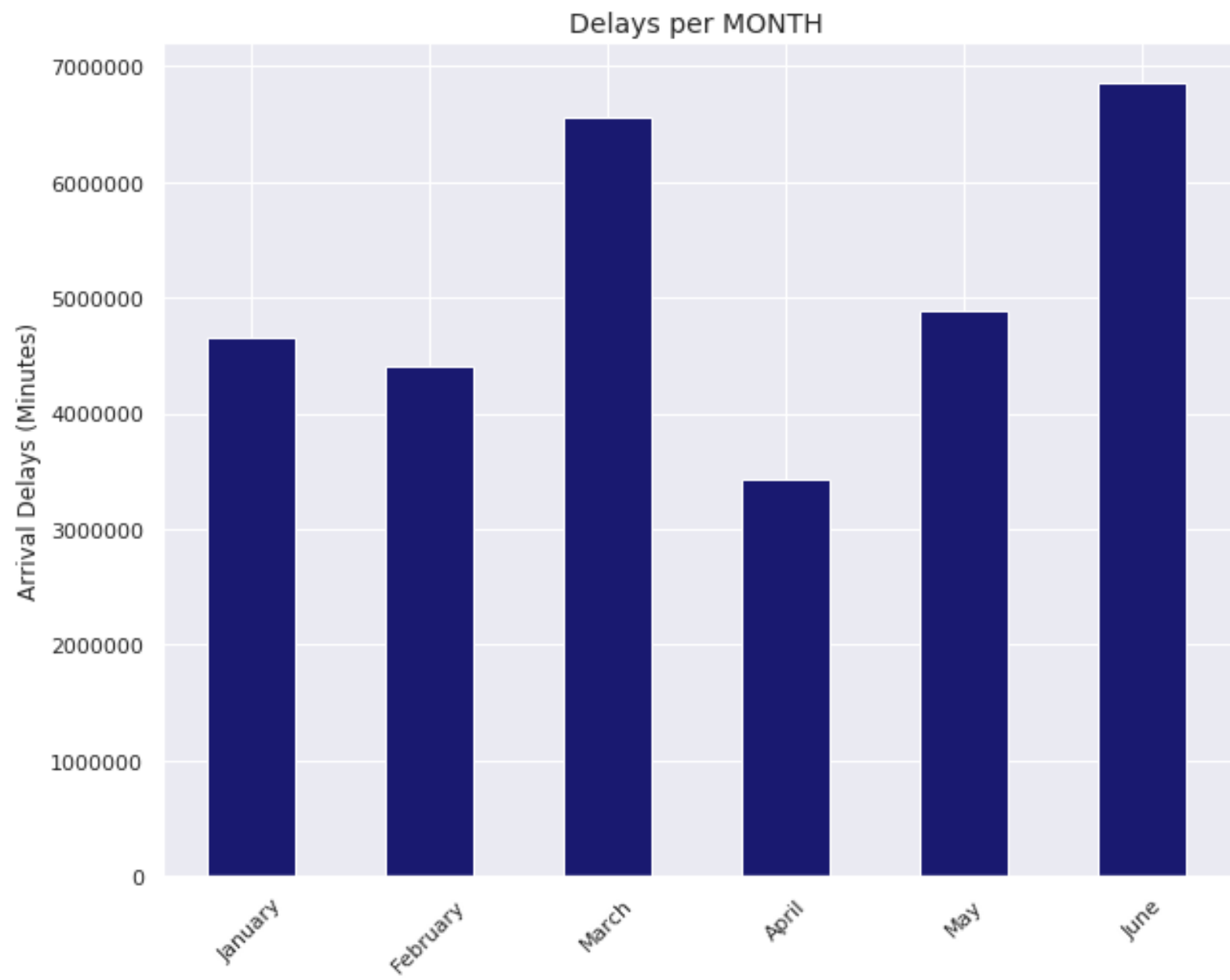


Les retards – Relation avec l'aéroport

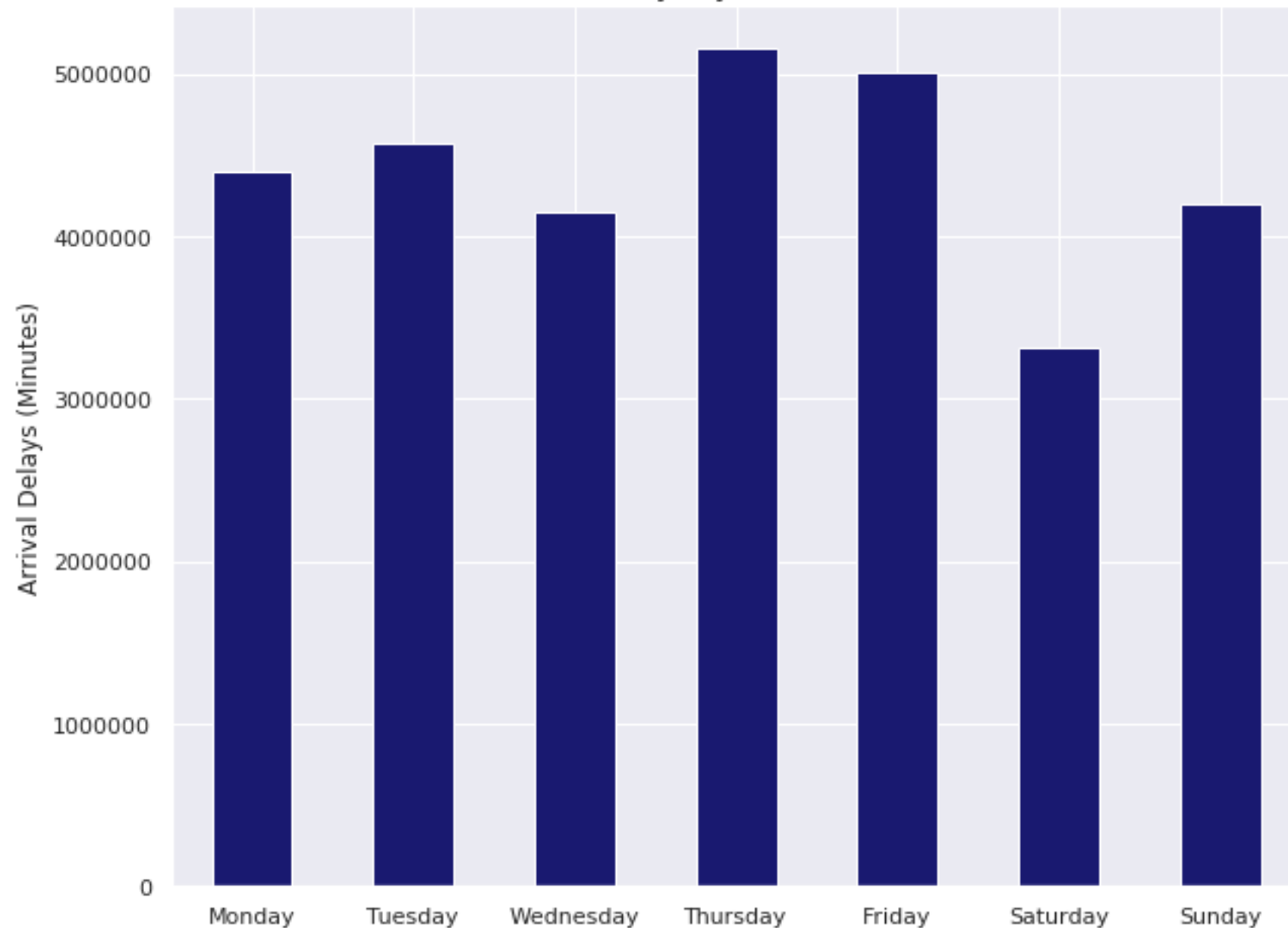
Delays by Airport



Les retards – Variations temporelles



Delays by WEEKDAY



Prédiction des retards

Modèle n°1: 1
compagnie, 1
aéroport

Modèle n°1: Jeu de données utilisé

- Mois de Janvier
- Aéroport d'origine: JFK
- Compagnie: B6
- Colonnes utilisées: DAY_OF_MONTH, DAY_OF_WEEK, MONTH, DEST, CRS_DEP_TIME, DEP_DELAY, CRS_ARR_TIME
- Algorithme utilisé: Régression linéaire

Modèle n°1: Prédiction des retards - Etapes

- Suppression:
 - $ARR_DELAY < 0$
 - CANCELLED
 - DIVERTED
- Encode (One hot encoding)
 - ORIGIN

Modèle n°1: Régression linéaire

```
# On crée un modèle de régression linéaire  
lr = linear_model.LinearRegression()  
  
# On entraîne ce modèle sur les données d'entraînement  
lr.fit(X_train_std,y_train)  
  
# On récupère l'erreur de norme 2 sur le jeu de données test comme baseline  
baseline_error = np.mean((lr.predict(X_test_std) - y_test) ** 2)  
  
print(baseline_error)
```

1486.1839609371352

Modèle n°1: Régression linéaire - Résultats

Score MSE du modèle: 1486.18

```
mse = mean_squared_error(pred_test_lr, y_test)
print("MSE =", mse)
```

```
MSE = 1486.1839609371357
```

Différence en minutes entre le délai prédit et le délai réel: 38.55 minutes

```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

```
'Delay = 38.55 min'
```


Modèle n°2: 1
compagnie, tous
les aéroports

Modèle n°2: Jeu de données utilisé

- Mois de Janvier
- Compagnie: B6
- Colonnes utilisées: DAY_OF_MONTH, DAY_OF_WEEK, MONTH, ORIGIN, DEST, CRS_DEP_TIME, DEP_DELAY, CRS_ARR_TIME
- Algorithmes utilisés: Régression linéaire, Régression Ridge, Lasso



Modèle n°2: La baseline - une régression classique

```
# On crée un modèle de régression linéaire  
lr = linear_model.LinearRegression()  
  
# On entraîne ce modèle sur les données d'entraînement  
lr.fit(X_train_std,y_train)  
  
# On récupère l'erreur de norme 2 sur le jeu de données test comme baseline  
baseline_error = np.mean((lr.predict(X_test_std) - y_test) ** 2)  
  
print(baseline_error)
```

2359.7400988796708

Modèle n°2: Régression Linéaire - Résultats

Score MSE du modèle: 2307.39

```
mse = mean_squared_error(y_test,y_pred)
print("MSE =", mse)
```

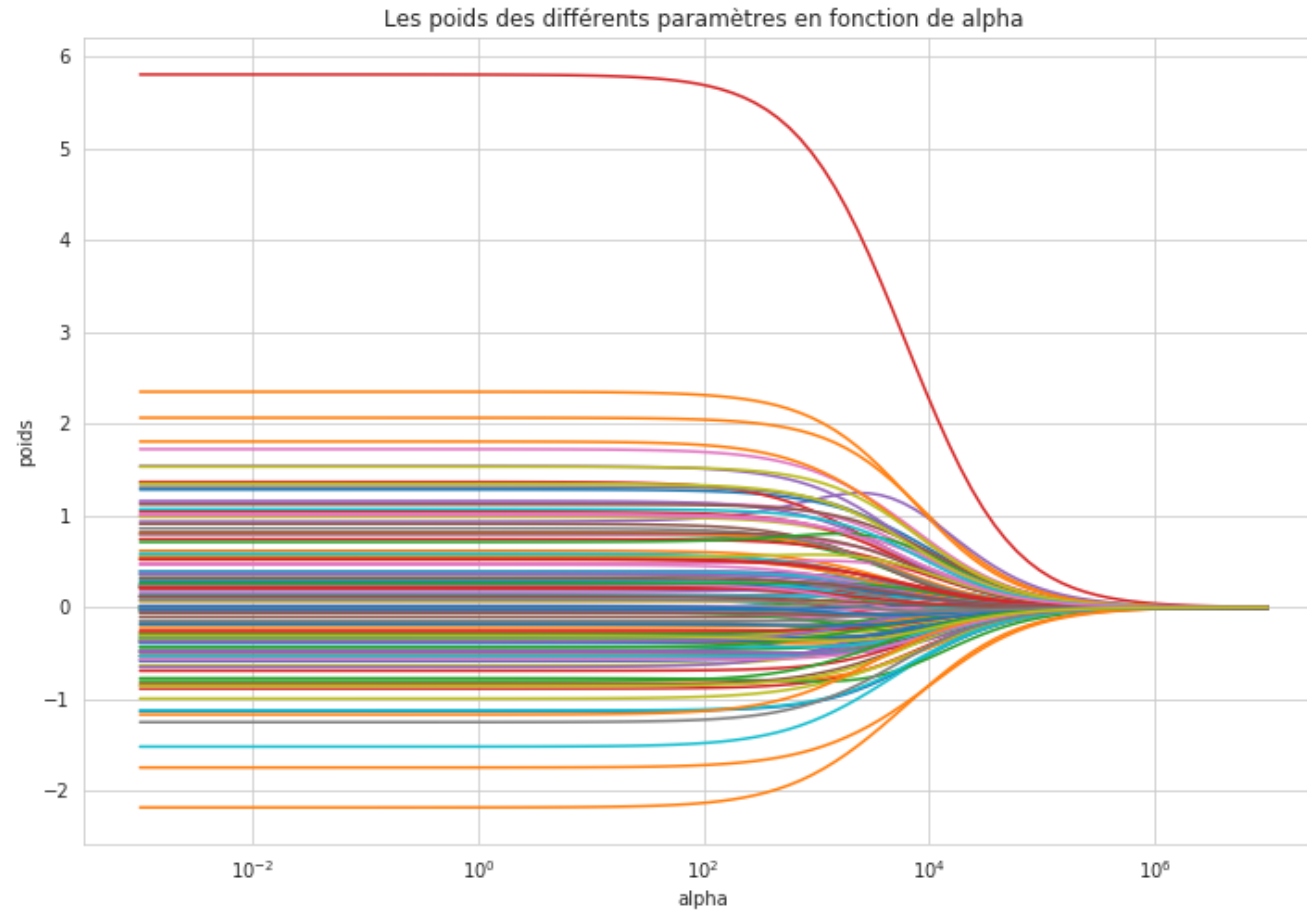
```
MSE = 2307.3883863761075
```

Différence en minutes entre le délai prédit et le délai réel: 48.04 minutes

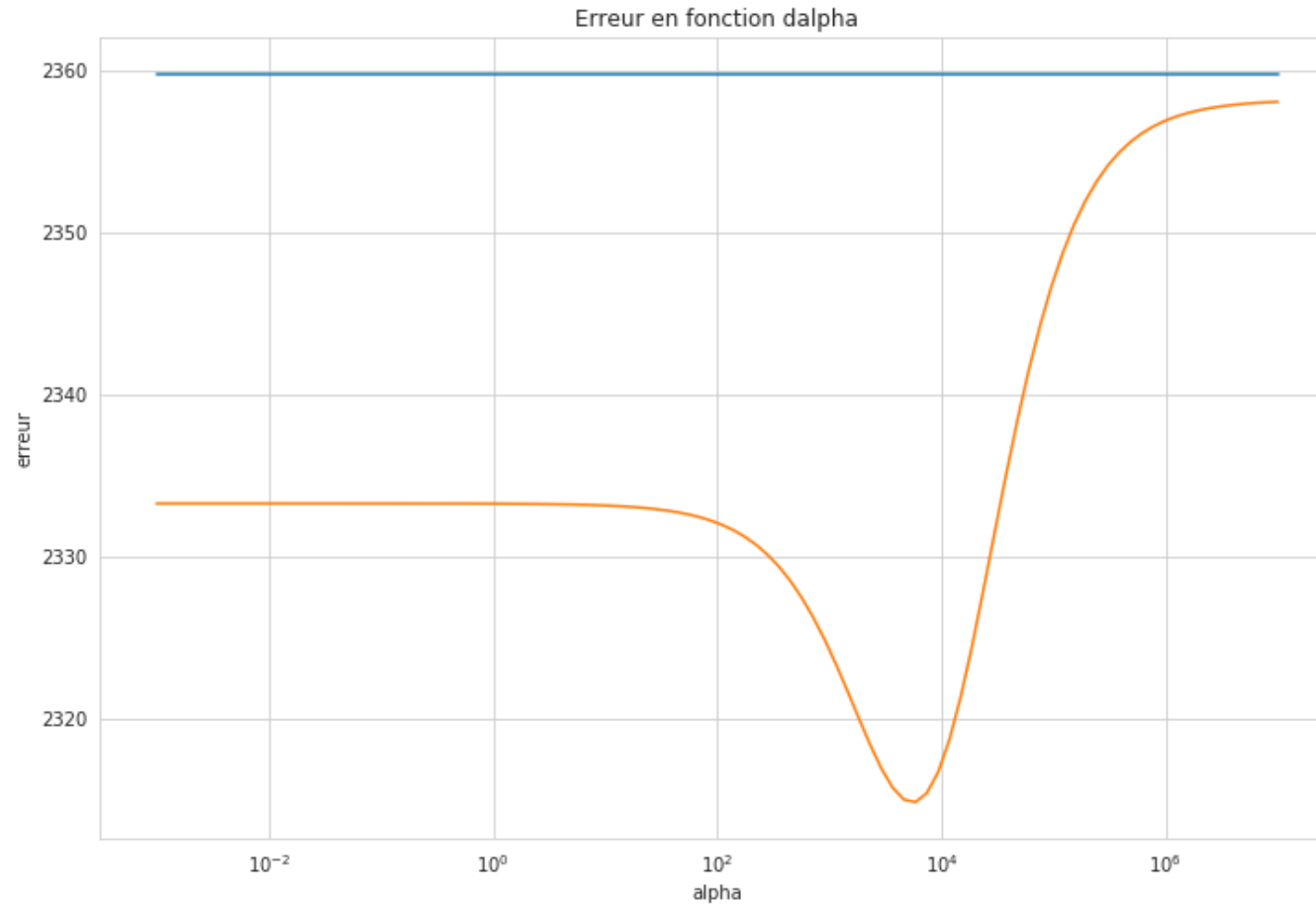
```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

```
'Delay = 48.04 min'
```

Modèle n°2: Application de la Régression Ridge – Poids des paramètres



Modèle n°2: Régression Ridge – Erreur quadratique



Modèle n°2: Régression Ridge – Cross validation

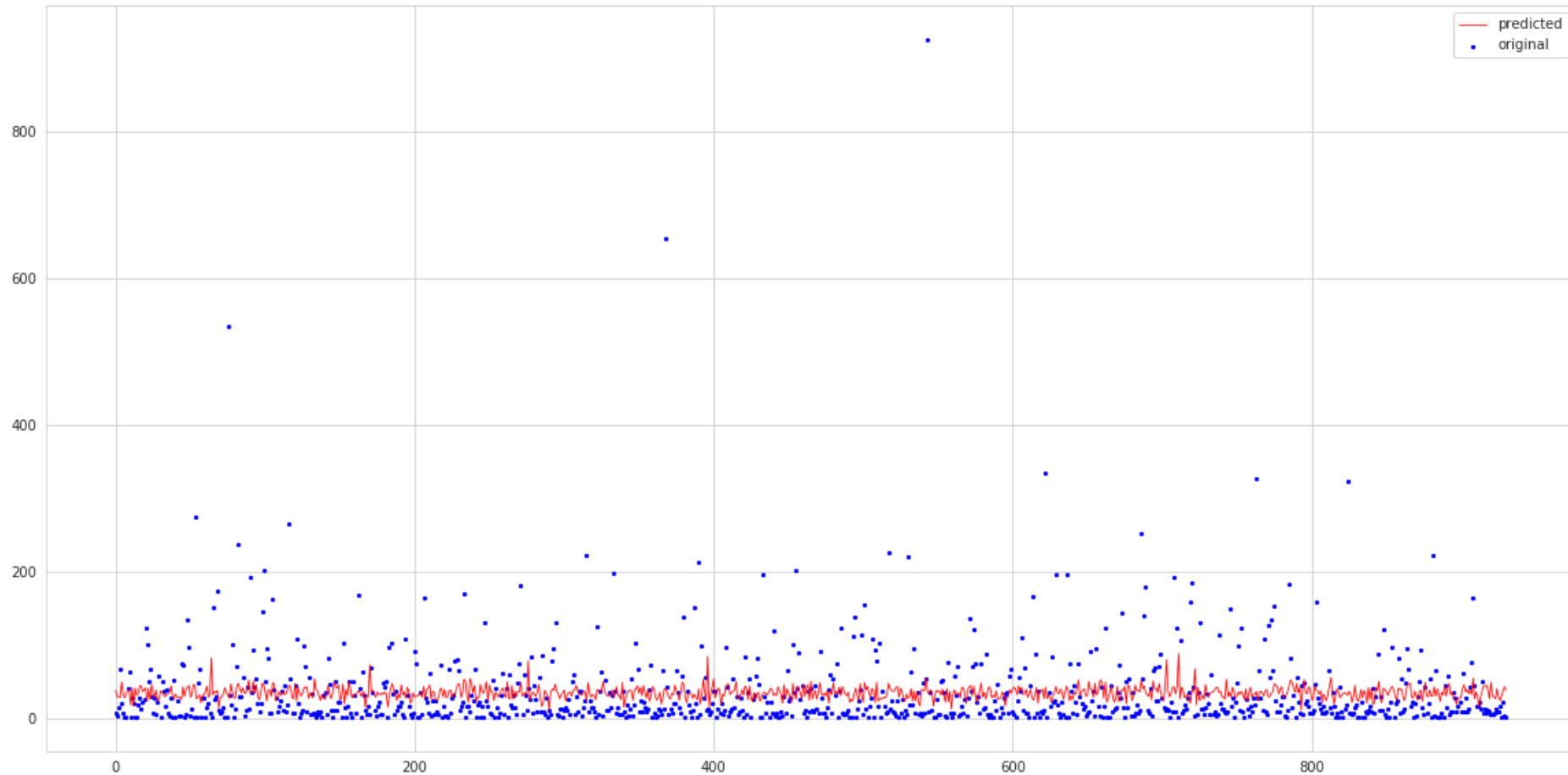
```
ridge_cv=RidgeCV(alphas=alphas, store_cv_values=True)
ridge_mod = ridge_cv.fit(X_train,y_train)
print(ridge_mod.alpha_)
```

50

```
y_pred = ridge_mod.predict(X_test)
score = ridge_mod.score(X_test,y_test)
mse = mean_squared_error(y_test,y_pred)
print("R2:{0:.3f}, MSE:{1:.2f}, RMSE:{2:.2f}"
      .format(score, mse, np.sqrt(mse)))
```

R2:0.026, MSE:2948.33, RMSE:54.30

Modèle n°2: Régression Ridge – Visualisation des résultats



Modèle n°2: Régression Ridge - Résultats

Score MSE du modèle: 2314.97

```
mse = mean_squared_error(y_test,y_pred)
print("MSE =", mse)
```

```
MSE = 2314.9686795106495
```

Différence en minutes entre le délai prédit et le délai réel: 48.11 minutes

```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

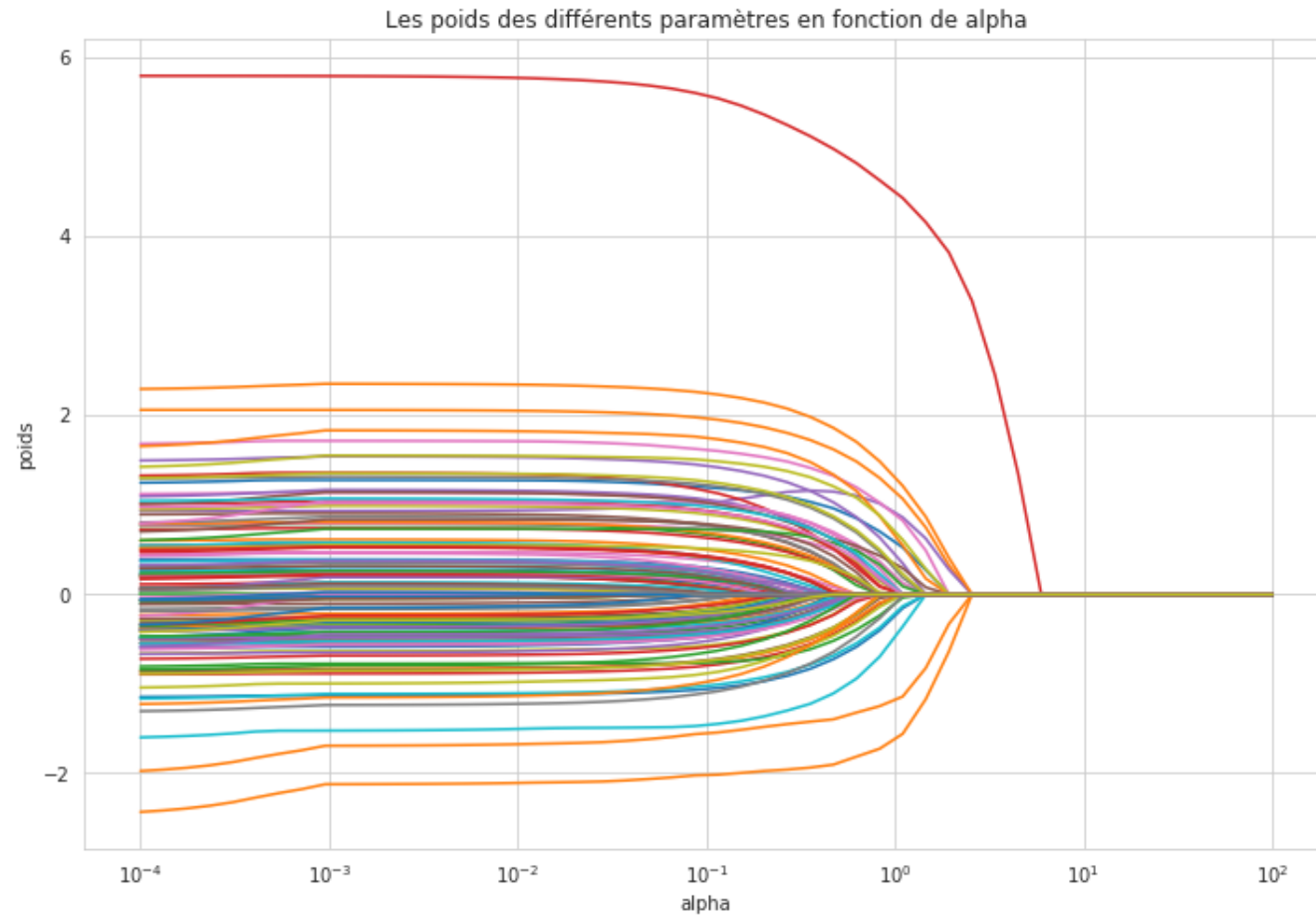
```
'Delay = 48.11 min'
```

Modèle n°2: Lasso – Test des différents hyperparamètres

```
n_alphas = 50
alphas = np.logspace(-5, 1, n_alphas)
lasso = linear_model.Lasso(fit_intercept=False)

coefs = []
errors = []
for a in alphas:
    lasso.set_params(alpha=a)
    lasso.fit(X_train, y_train)
    coefs.append(lasso.coef_)
    errors.append([baseline_error, np.mean((lasso.predict(X_test) - y_test) ** 2)])
```

Modèle n°2: Lasso – Poids des paramètres



Modèle n°2: Lasso – Cross validation

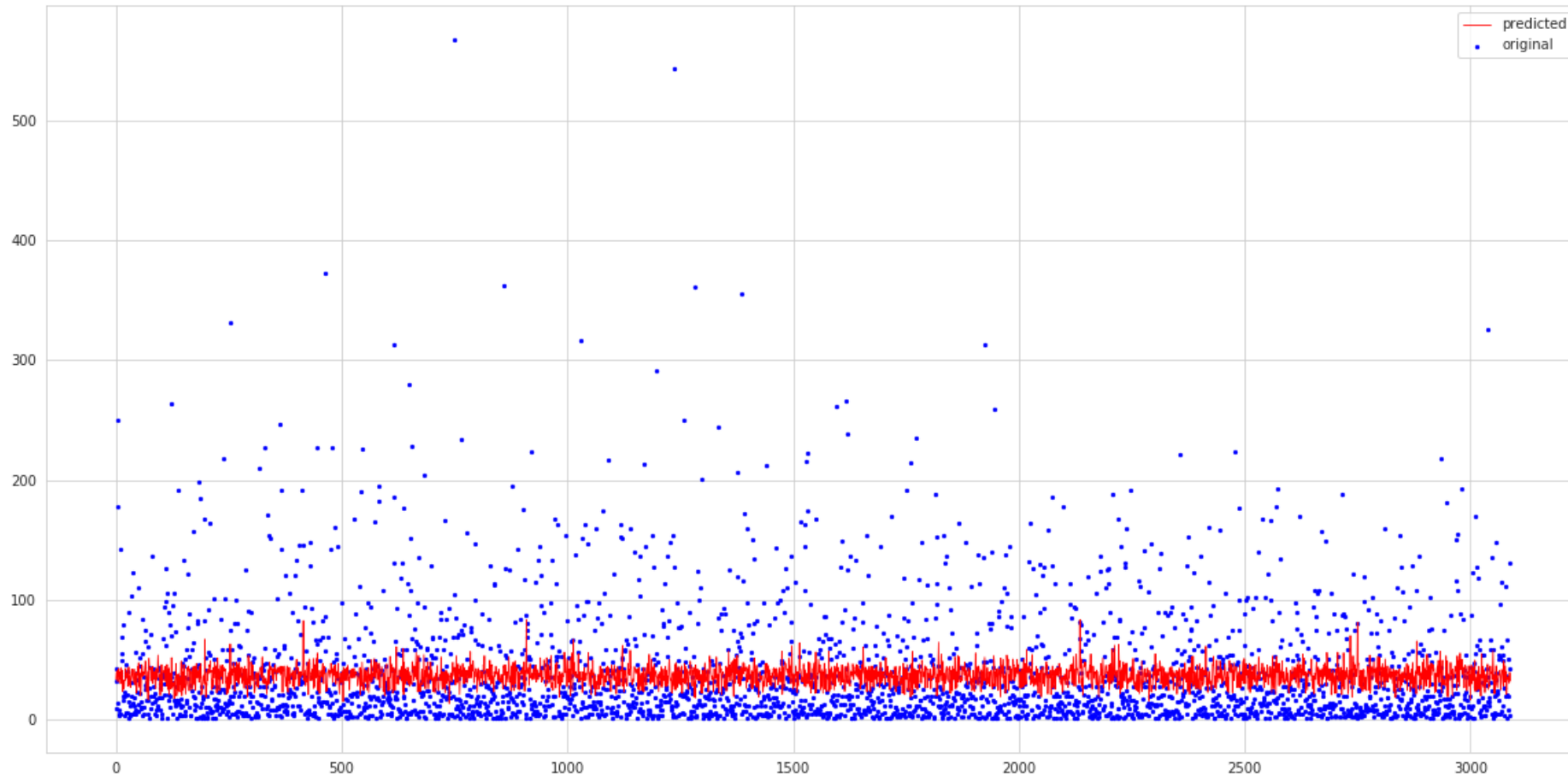
```
alphas = [0.1, 0.3, 0.5, 0.8, 1]
lassocv = LassoCV(alphas=alphas, cv=5).fit(X_train, y_train)
print(lassocv.alpha_)
```

0.1

```
score = lassoCV.score(X_train, y_train)
y_pred = lassoCV.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print("R2:{1:.3f}, MSE:{2:.2f}, RMSE:{3:.2f}"
      .format(lassocv.alpha_, score, mse, np.sqrt(mse)))
```

R2:0.025, MSE:2963.92, RMSE:54.44

Modèle n°2: Lasso – Visualisation des résultats



Modèle n°2: Lasso - Résultats

Score MSE du modèle: 2307.39

```
mse = mean_squared_error(y_test,y_pred)
print("MSE =", mse)
```

```
MSE = 2307.3883863761075
```

Différence en minutes entre le délai prédit et le délai réel: 48.04 minutes

```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

```
'Delay = 48.04 min'
```

Modèle n°3:
Toutes les
compagnies, un
aéroport

Modèle n°3: Jeu de données utilisé

- Mois de Janvier
- Aéroport: JFK
- Colonnes utilisées: DAY_OF_MONTH, DAY_OF_WEEK, MONTH, UNIQUE_CARRIER, DEST, CRS_DEP_TIME, DEP_DELAY, CRS_ARR_TIME
- Algorithmes utilisés: Régression Ridge, Lasso

Modèle n°3: Régression Ridge - Résultats

Score MSE du modèle: 3610.58

```
mse = mean_squared_error(y_test,y_pred)
print("MSE =", mse)
```

```
MSE = 3610.5792800107915
```

Différence en minutes entre le délai prédit et le délai réel: 60.09 minutes

```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

```
'Delay = 60.09 min'
```

Modèle n°3: Lasso - Résultats

Score MSE du modèle: 3594.89

```
mse = mean_squared_error(y_test, y_pred)
print("MSE =", mse)
```

```
MSE = 3594.8869011801394
```

Différence en minutes entre le délai prédit et le délai réel: 59.96 minutes

```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

```
'Delay = 59.96 min'
```

Evaluation des modèles

Modèle n°3: Comparaison avec un modèle aléatoire - Résultats

Score MSE du modèle: 3676.86

```
# Evaluate  
mse = mean_squared_error(y_test, y_pred_dum)  
print("MSE =", mse)
```

```
MSE = 3676.85882056067
```

Différence en minutes entre le délai prédit et le délai réel: 60.64 minutes

```
'Delay = {:.2f} min'.format(np.sqrt(mse))
```

```
'Delay = 60.64 min'
```

Choix du modèle de prédiction

Modèle	RMSE	Retard
Ridge	3610.58	60.09
Lasso	3594.89	59.96
Aléatoire	3676.86	60.64

- Les modèles appliqués ne donnent pas de résultats satisfaisants.
- La régression Ridge fait mieux que Lasso et sera donc utilisée comme modèle pour la prédiction du retard des avions.

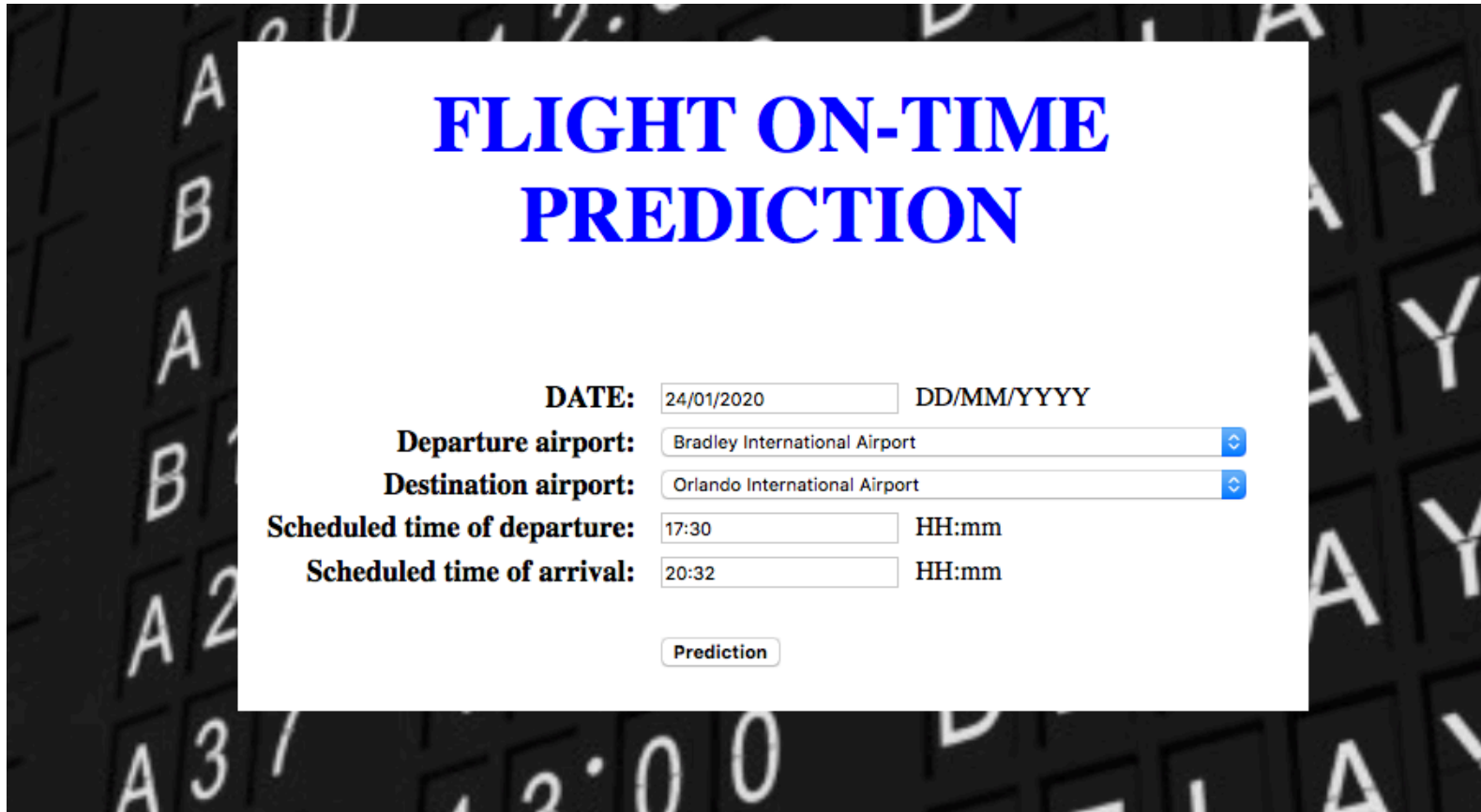
Application de prédiction de retard des avions

Application web

Application disponible ici -> <https://flights-delay-pred.herokuapp.com>



Application web



**FLIGHT ON-TIME
PREDICTION**

DATE: DD/MM/YYYY

Departure airport:

Destination airport:

Scheduled time of departure: HH:mm

Scheduled time of arrival: HH:mm

Application web

Your predicted delay is: 30 minutes

[Get another flight delay prediction](#)

Conclusion – Améliorations possibles

- ❑ Inclure plus de données venant d'autres années.
- ❑ Les données pourraient être complétées par des données sur les conditions météorologiques et des données de maintenance des avions/aéroports afin de donner des résultats plus précis.

Merci